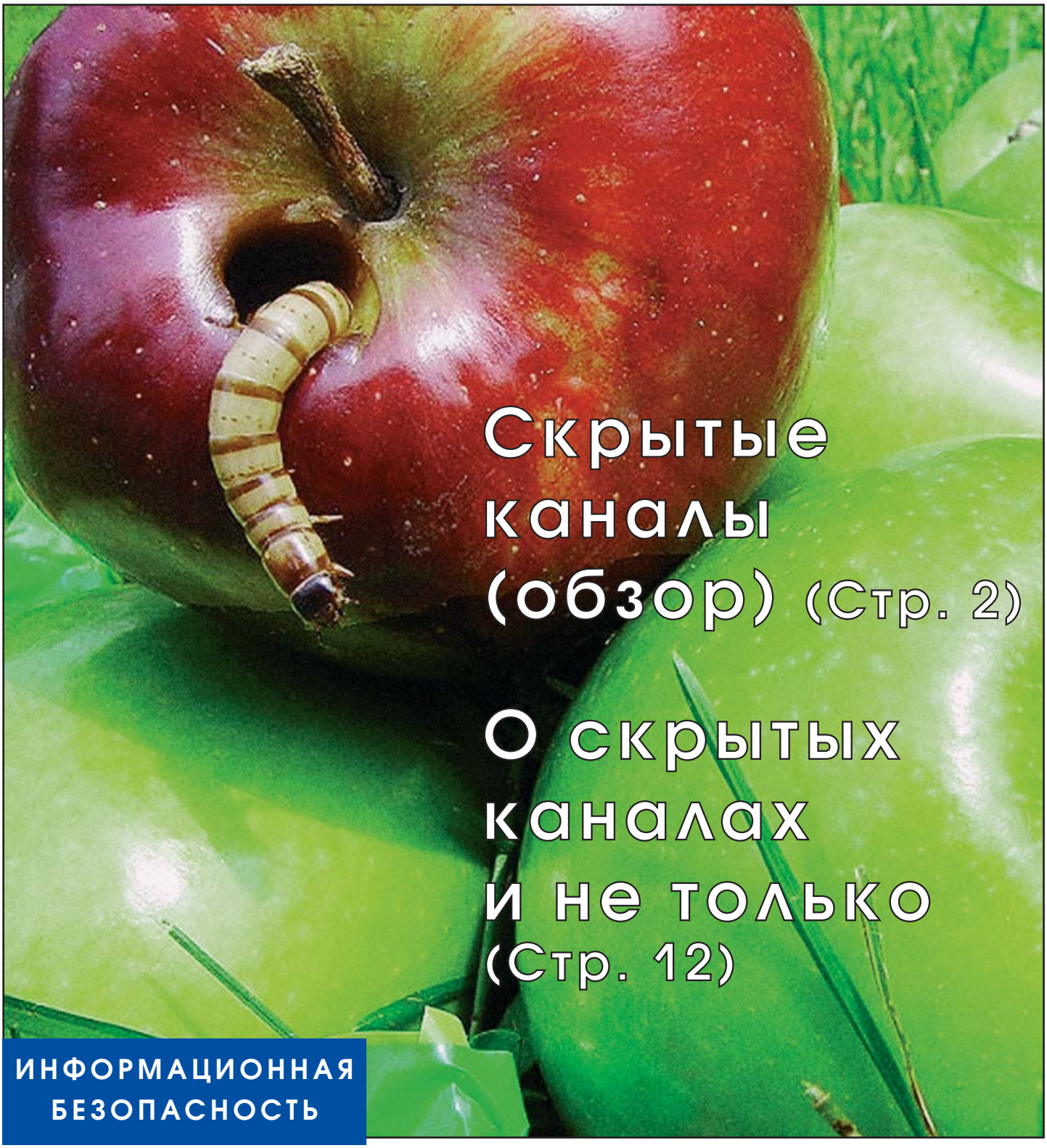
Jet Info

ИНФОРМАЦИОННЫЙ БЮЛЛЕТЕНЬ № 11 (114)/2002



Скрытые каналы (обзор) (Стр. 2)

О скрытых каналах и не только (Стр. 12)

ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ

Скрытые каналы (обзор)

Работа поддержана грантом РФФИ 01-01-00895

Тимонина Е. Е.

СОДЕРЖАНИЕ

1. Введение 3

2. Информационный поток 3

Пример 1.
Пример 2.
Пример 3.

3. Политики безопасности 5

4. Примеры скрытых каналов 7

5. Модели скрытых каналов и их анализ 8

6. Борьба со скрытыми каналами 10

Литература 11

1. Введение

Попытки скрыть сам факт передачи информации имеют длинную историю. Способы сокрытия самого факта передачи информации получили название стеганография. Исторически для стеганографии применялись «невидимые» чернила, точечные фотовставки и т.д. Данное направление получило вторую жизнь в наше время в связи с широким использованием сетей передачи данных. Чтобы выделить методы стеганографии, связанные с электронным представлением данных, появился термин компьютерная стеганография. Однако в работе Шнайера [22] стеганографические способы передачи по каналам связи получили название потайных каналов (subliminal channels). Наряду с этим появился термин «скрытый канал» (covert channel). Впервые понятие скрытого канала было введено в работе Лэмпсона [16] в 1973 году. Канал называется скрытым, если он не проектировался, не предполагался для передачи информации в электронной системе обработки данных. Таким образом, термин скрытые каналы больше относится к внутри компьютерным телекоммуникациям. В работе Тсаи [23] дано следующее определение скрытого канала. Если нам дана модель не дискреционной политики безопасности M и ее имплементация $I(M)$ в операционной системе, то любая потенциальная связь между двумя субъектами $I(Sh)$ и $I(Si)$ в $I(M)$ называется скрытым каналом, если эта связь между субъектами Sh и Si в модели M не разрешена.

Поскольку все приведенные выше термины, касающиеся скрытой передачи информации, отличаются нюансами приложений, мы будем без ограничения общности способы скрытой передачи информации называть скрытыми каналами. Возвращаться к исходным терминам мы будем в тех особых случаях, когда выделение соответствующих каналов вытекает из контекста.

Во всех случаях скрытых каналов необходимо уточнять понятие канала связи. В связи с этим в следующем разделе 2 мы приводим различные определения, которые порождают различные понимания скрытых каналов, использованных в этом обзоре.

В разделе 3 мы раскрываем связь различных политик безопасности и скрытых каналов. Выявление этой связи позволяет определить, от чего и как защищают те или иные механизмы реализации различных политик безопасности, и в каких случаях вопрос о существовании скрытых каналов требуется решать дополнительно.

4 раздел посвящен примерам скрытых каналов в различных трактовках этого термина.

Раздел 5 посвящен задачам анализа скрытых каналов. В нем приводятся методы выявления скрытых каналов при анализе компьютерных систем и программного обеспечения. В частности, рассматривается известный метод зависимостей и метод матрицы общих ресурсов.

В разделе 6 приводятся некоторые данные о противодействии стеганографическим схемам в некоторых традиционных видах носителей.

2. Информационный поток

Простейший подход к определению информационного потока можно найти в TCSEC [11] («Оранжевая книга», 1985 г.) и на русском языке в работе А.Грушо и Е.Тимониной [1]. Если осуществляется доступ на чтение (read) субъекта S к объекту O, то поток информации идет от O к S. Если S имеет доступ на запись (write) к O, то информационный поток направлен от S к O. Транзитивное замыкание цепочек доступа (даже без учета времени) представляет собой сложный информационный поток.

Пример 1.

Например, документ D записан в файл O пользователем U_1, а пользователь U_2 прочитал документ D в файле O. Таким образом информация, содержащаяся в документе D, перешла от пользователя U_1 к пользователю U_2 с помощью двух информационных потоков:

- U_1 организовал информационный поток от себя к O с помощью операции write;

- U_2 организовал информационный поток от O к себе с помощью операции read.

Более привычное определение информационного потока вводится через среднюю взаимную информацию (см. работы Шеннона [8]). Как определено в работе [1] объект O в информационной системе представляет собой конечное множество допустимых записей в данном языке, а состоянием объекта O является конкретная запись из этого конечного множества, которая находится в информационной системе в данный момент времени с именем O. Пусть X и Y два объекта в информационной системе и предположим, что в данный момент времени состояния объектов X и Y определяются совместным распределением вероятностей $P(x, y)$ на конечном множестве пар (x, y), $x \in X$, $y \in Y$, $P(x,y) \geq 0$, $\sum_{y \in Y,\, x \in X} P(x,y) = 1$. Обозначим $P_X(x) = \sum_{y \in Y} P(x,y)$ и аналогично $P_Y(y) = \sum_{x \in X} P(x,y)$. Средней взаимной информацией объектов X и Y называется величина $I(X,Y) = \sum_{y \in Y,\, x \in X} P(x,y) \log_2 \frac{P(x,y)}{P_X(x) P_Y(y)}$.

Определение. *Информационный поток между объектами X и Y существует, если средняя взаимная информация $I(X,Y) > 0$.*

Можно доказать, что существование информационного потока эквивалентно условию, что существует пара (x,y) такая, что $P(x,y) \neq P_X(x) P_Y(y)$.

Ясно, что если существует информационный поток от X к Y, то существует такой же информационный поток от Y к X. В компьютерной безопасности данный подход к описанию информационных потоков предложен в работе Деннинг [10].

Пример 2.

Пусть файл O_1 является копией файла O_2. Тогда $P(x,y) = 0$ при $x \neq y$. Если вероятность состояния x объекта O_1 не равна 0, и вероятность состояния y объекта O_2 не равна 0, то из эквивалентного определения информационного потока следует, что существует информационный поток от объекта O_1 к объекту O_2.

Другие примеры простейших потоков данного типа в программах можно найти в [1]. Отметим, что данное определение потока эквивалентно заданию совместного распределения на множествах состояний объектов X и Y при условии, что меры, индуцируемые на X и Y, не являются независимыми.

Рассмотрим еще одно определение информационного потока, приведенное в [13, 19]. В этих работах для анализа скрытых каналов вводится понятие зависимости. С точки зрения нашего анализа любая зависимость порождает канал передачи данных. Поэтому мы рассматриваем понятие зависимостей как один из способов определения информационного потока.

Определение. *Информационным потоком от объектов $\{S\}$ к объекту T можно считать тройку $(T, \{S\}, G)$, где T меняет свое состояние, если $\{S\}$ меняет свое состояние при условии, что логическое выражение G принимает значение истина.*

Пример 3.

При вхождении в систему пользователь называет свое имя $И$ и вводит свой пароль $П$. Данный пароль после шифрования преобразуется в запись $К(П)$, которая сравнивается со значением в таблице паролей, соответствующей данному имени $И$ пользователя. В данном случае значение переменной T, равное 1, если пользователю разрешено войти в систему, и 0 — в противном случае, зависит от введенных переменных $И$ и $П$ при выполнении логического условия, что в соответствующей строчке таблицы паролей $К(П)$ введенного пароля совпадает с имеющейся там записью. В данном случае переменная S принимает значение имени пользователя $И$, переменная S_1 принимает значение пароля $К$, а логическое условие G соответствует совпадению записей в таблице паролей.

К этому определению относятся все виды функциональных связей, в которых значение T есть функция от некоторого набора переменных, куда входит $\{S\}$.

Обобщением данной схемы является модель информационного потока как конечного автомата, в котором источник сообщения посылает входное слово на вход автомата, а получатель сообщения видит выходную последовательность автомата [6]. Следующими обобщениями являются модель канала как вероятностного автомата и модель детерминированного автомата со случайным входом.

3. Политики безопасности

Будем следовать общепринятому определению политики безопасности (ПБ), приведенному в стандарте TCSEC [11].

Определение. *Политика безопасности это набор норм, правил и практических приемов, которые регулируют управление, защиту и распределение ценной информации.*

Многие политики безопасности выражаются через информационные потоки. Например, все информационные потоки в системе (в том числе потенциальные) делятся на два непересекающихся подмножества: разрешенные и неразрешенные потоки. Тогда система защиты должна обеспечивать поддержку разрешенных потоков и препятствовать запрещенным потокам. К политикам такого класса относится многоуровневая политика (MLS).

Многоуровневая политика безопасности (политика MLS) принята всеми развитыми государствами мира. В секретном повседневном делопроизводстве госсектор России также придерживается этой политики.

Решетка ценностей *SC* является основой политики MLS. Линейно упорядоченное множество грифов секретности «несекретно» < «секретно» < «совершенно секретно» является простейшим примером такой решетки ценности. В более общем случае к грифам конфиденциальности добавляются подмножества тематических категорий из заданного набора категорий. В этом случае также получается решетка ценностей, в которой некоторые элементы упорядочены. Например, приведем сравнение двух элементов такой решетки: «секретно, кадры, финансы» < «совершенно секретно, кадры, финансы, материальное обеспечение». Классификация информационных ресурсов - это отображение c множества объектов системы O в множество узлов SC решетки ценности. То есть каждый объект системы классифицируется уровнем секретности и множеством тематических категорий. Отображение c: $O \to SC$ считается заданным. Если $c(Y) \geq c(X)$, то Y – более ценный объект, чем X.

Определение. *Политика MLS считает информационный поток $X \to Y$ разрешенным тогда и только тогда, когда $c(Y) \geq c(X)$ в решетке SC.*

Таким образом политика MLS имеет дело с множеством информационных потоков в системе и делит их на разрешенные и неразрешенные очень простым условием. Однако эта простота касается информационных потоков, которых в системе огромное количество. Поэтому приведенное выше определение неконструктивно. Хотелось бы иметь конструктивное определение на языке доступов. Рассмотрим класс систем с двумя видами доступов read и write (хотя могут быть и другие доступы, но они либо не определяют информационный поток, либо выражаются через write и read). Пусть процесс S в ходе решения своей задачи последовательно обращается к объектам O_1, O_2,..., O_l (некоторые из них могут возникнуть в ходе решения задачи). Пусть

$$S \xrightarrow{r} O_{i_1},\ldots,S \xrightarrow{r} O_{i_k},\ S \xrightarrow{w} O_{j_1},\ldots,S \xrightarrow{w} O_{j_{l-k}} \qquad (1)$$

Тогда из определения MLS следует, что при выполнении условий

$c(S) \geq c(O_{i_t})$, $t = 1,\ldots,k$, соответствующие потоки информации, определяемые доступом read, будут идти в разрешенном политикой MLS направлении, а при

$c(S) \leq c(O_{i_t})$, $t = 1,\ldots,k$, потоки, определяемые доступом write, будут идти в разрешенном направлении. Таким образом, в результате выполнения задачи процессом S, информационные потоки, с ним связанные, удовлетворяют политике MLS. Такого качественного анализа оказывается достаточно, чтобы классифицировать почти все процессы и принять решение о соблюдении или нет политики MLS. Если где-то политика MLS нарушается, то соответственный доступ не разрешается. Причем разрешенность цепочки (1) вовсе не означает, что субъект S не может создать объект O такой, что $c(S) > c(O)$. Однако он не может писать туда информацию. При передаче управления поток информации от процесса S или к нему прерывается (хотя в него другие процессы могут записывать или считывать информацию как в объект). При этом, если правила направления потока при read и write выполняются, то MLS соблюдается, если нет, то соответствующий процесс не получает доступ. Таким образом, мы приходим к управлению потоками через контроль доступов. В результате для определенного класса систем получим конструктивное описание политики MLS.

Определение. *В системе с двумя доступами read и write политика MLS определяется следующими правилами доступа*

$$X \xrightarrow{r} Y \iff c(X) \geq c(Y),$$
$$X \xrightarrow{w} Y \iff c(X) \leq c(Y).$$

Структура решетки очень помогает организации поддержки политики MLS. В самом деле, пусть имеется последовательная цепочка информационных потоков

$O_1 \to O_2 \to O_3 \to \ldots \to O_k$.

Если каждый из потоков разрешен, то свойства решетки позволяют утверждать, что разрешен

сквозной поток $O_1 \rightarrow O_k$. Действительно, если информационный поток на каждом шаге разрешен, то $c(O_{i+1}) \geq c(O_k)$, тогда по свойству транзитивности решетки $c(O_1) \leq c(O_k)$, то есть сквозной поток разрешен.

MLS политика в современных системах защиты реализуется через мандатный контроль (или, также говорят, через мандатную политику). Устройство мандатного контроля, удовлетворяющее некоторым дополнительным требованиям, называется монитором обращений. Мандатный контроль еще называют обязательным, так как через него проходит каждое обращение субъекта к объекту, если субъект и объект находятся под защитой системы безопасности. Организуется мандатный контроль следующим образом. Каждый объект O имеет метку с информацией о классе $c(O)$. Каждый субъект также имеет метку, содержащую информацию о том, какой класс доступа $c(S)$ он имеет. Мандатный контроль сравнивает метки и удовлетворяет запрос субъекта S к объекту O на чтение, если $c(S) \geq c(O)$ и удовлетворяет запрос на запись, если $c(S) \leq c(O)$. Тогда согласно изложенному выше мандатный контроль реализует политику MLS. Однако метки конфиденциальности это не единственный способ управления информационными потоками в компьютерных системах.

В системе с многоуровневой политикой безопасности, в которой информационные потоки сведены к доступам, возможны потоки более общего типа из рассмотренных в п. 2, которые могут нарушать политику безопасности MLS. Например, любой информационный поток между несравнимыми узлами решетки или сверху вниз, который существует, но не выражается через доступы read и write, будет нарушать политику безопасности MLS даже при корректной реализации мандатного контроля доступов. В простейшем случае, к которому мы будем апеллировать дальше, если система разделена, по крайней мере, на два уровня High и Low и в системе принята многоуровневая политика безопасности [1], разрешающая информационные потоки снизу вверх (от Low к High) и запрещающая потоки сверху вниз, то нарушитель может использовать скрытый канал для передачи информации от программно-аппаратного агента в среде High к программно-аппаратному агенту в среде Low. При этом скрытый канал должен защищать нарушителя от системы защиты, поддерживающей многоуровневую политику, основанную на определении потоков через доступы read и write. То есть скрытый канал должен быть невидим монитору обращений, системе аудита, аналитику, исследующему защищенность системы и т.д.

Потоки в MLS разрешены только между сравнимыми узлами снизу вверх. Данная политика защищает конфиденциальность информации. Точно так же, как многоуровневая политика, определяется политика защиты целостности Байба [9], только разрешенными в данной политике являются все потоки между сравнимыми узлами, направленные вниз.

Предположим, что опасности для нарушения секретности не существует, а единственная цель политики безопасности - защита от нарушений целостности информации. Пусть, по-прежнему, в информацию внесена решетка ценностей *SC*. В этой связи любой информационный поток $X \rightarrow Y$ может воздействовать на целостность объекта Y. Если в Y более ценная информация, чем в X, то такой поток при нарушении целостности Y принесет более ощутимый ущерб, чем поток в обратном направлении от более ценного объекта Y к менее ценному X. Байба [9] предложил в качестве политики безопасности для защиты целостности следующее.

Определение. *В политике Байба информационный поток $X \rightarrow Y$ разрешен тогда и только тогда, когда $c(Y) \leq c(X)$.*

Можно показать, что в широком классе систем эта политика эквивалентна следующей.

Определение. *Для систем с доступами write и read политика Байба разрешает доступ в следующих случаях:*

$$S \xrightarrow{r} O \iff c(S) \leq c(O),$$
$$S \xrightarrow{w} O \iff c(S) \geq c(O).$$

Очевидно, что для реализации этой политики также подходит мандатный контроль. Точно так же как и раньше при выполнении мандатного контроля доступов read и write возможно нарушение политики Байба с помощью скрытых каналов (информационных потоков более общего типа).

Помимо указанных политик следует назвать класс политик защиты связи, в которых информационный поток, передаваемый от отправителя к получателю, не должен быть перехвачен или искажен при различных допущениях относительно возможностей противника по искажению информационных потоков или по перехвату части передаваемой информации, или наоборот, по попыткам вклиниться в передаваемый информационный поток. Сюда следует отнести ряд стеганографических схем, в которых основная задача создать информационный поток, «невидимый» для наблюдателя, с определенным набором возможностей.

Еще одним примером [4] является ситуация, когда производитель продает пользователю компьютерную систему для обработки данных, при этом производитель встроил программно-аппаратного агента для анализа данных, которые обрабаты-

ваются покупателем. Данная система может быть сделана таким образом, что программно-аппаратный агент соответствует уровню High, а легальный вычислительный процесс проходит на уровне Low. Для передачи агентом информации во вне системы необходимо построить скрытый канал между верхним и нижним уровнями с выходом во внешнюю среду (например, в Интернет). Аналогично агент должен получать инструкции с нижнего уровня скрытно, поскольку входные сообщения для легального вычислительного процесса и агента приходят по одному каналу.

Суммируем кратко выводы раздела 3. Для поддержки политики безопасности используются механизмы защиты, препятствующие нарушению политики безопасности. Одним из способов нарушения политики безопасности является создание скрытых информационных потоков, не выявляемых системами защиты. В случае многоуровневой политики скрытые каналы передают информацию с верхних уровней конфиденциальности на нижний уровень так, чтобы механизмы защиты не могли препятствовать нарушению политики защиты конфиденциальности. В политике Biba скрытый канал с нижнего уровня на верхний может передать команду "Троянскому коню" на уничтожение или модификацию информационных ресурсов, целостность которых защищает данная политика.

В связи с этим возникла проблема анализа скрытых каналов всюду, где возникают ограничения на информационные потоки. Любой такой анализ предполагает решение четырех взаимосвязанных задач:

1) Выявление скрытых каналов;
2) Оценка пропускной способности скрытых каналов и оценка опасности, которую несет их скрытое функционирование;
3) Выделение сигнала или получение какой-нибудь информации, передаваемой по скрытым каналам;
4) Противодействие реализации скрытого канала вплоть до его уничтожения.

4. Примеры скрытых каналов

Традиционно [19] скрытые каналы характеризуются как каналы по памяти или каналы по времени. В работе Кемерера [15] определяются скрытые каналы по памяти как такие каналы, в которых информация передается через доступ отправителя на запись и получателя на чтение к одним и тем же ресурсам или объектам. Скрытый канал по времени характеризуется доступом отправителя и получателя к одному и тому же процессу или изменяемому во времени атрибуту.

Как и прежде будем полагать, что система разделена, по крайней мере, на два уровня High и Low и в системе принята многоуровневая политика безопасности [1], разрешающая информационные потоки снизу вверх (от Low к High) и запрещающая потоки сверху вниз. Нарушитель может использовать скрытый канал для передачи информации от программно-аппаратного агента в среде High к программно-аппаратному агенту в среде Low. При этом скрытый канал должен защищать нарушителя от системы защиты, поддерживающей многоуровневую политику.

Простейшим скрытым каналом по памяти является возможность показа на уровне Low названий директорий и файлов, созданных на уровне High. В данном случае информация может передаваться в именах файлов, которые выбираются в соответствии с заранее условленным кодом, в атрибутах файлов, в которых информация может кодироваться, размерами файлов, датами изменения файлов и т.д. И, наконец, существование файла с данным названием несет бит информации с верхнего уровня на нижний.

Другим примером канала по памяти является кодирование информации в сохраняемых настройках каких-либо ресурсов общего пользования субъектов уровней High и Low. Настройки, проведенные на уровне High, доступны наблюдению на уровне Low и, следовательно, могут нести информацию, выраженную заранее условленным кодом.

Скрытые каналы по времени впервые стали серьезно рассматриваться с 1976 г., когда один из создателей защищенной операционной системы Multics Миллен [17] продемонстрировал своим коллегам скрытый канал по времени, реализованный на изолированных машинах High и Low. Обе машины были подсоединены к некоторым общим ресурсам ROM, других каналов или связей между ними не было. В подсистемах High и Low находились "Троянские кони". На уровне High "Троянский конь" при нажатии букв на клавиатуре модулировал специаль-

ным кодом интервалы времен занятости библиотеки ROM. Время занятости библиотеки верхним уровнем сканировалось запросами в библиотеку "Троянским конем" нижнего уровня. Получившийся скрытый канал по времени позволял в реальном времени печатать информацию, получаемую через скрытый канал с клавиатуры подсистемы уровня High.

Рассмотрим еще один пример скрытого канала по времени. Пусть в программно-аппаратной схеме, реализующей интерфейс RS 232 между Low и High, нет передатчика на уровне High и нет приемника на уровне Low. Вместе с тем для передачи байт с нижнего уровня на верхний машина верхнего уровня выставляет сигнал готовности к приему информации. Очередной байт передается только тогда, когда выставлен сигнал готовности приема. Тогда задержка в выставлении сигнала после очередного переданного байта считается таймером на нижнем уровне и может таким образом передавать информацию от программно-аппаратного агента на верхнем уровне к программно-аппаратному агенту на нижнем уровне. Для этого агент на верхнем уровне кодирует сообщение различными по длине интервалами задержки выставления сигнала, а агент на нижнем уровне считывает эти сообщения с помощью таймера.

Хорошо известен пример построения скрытого канала в работе Шнайера [22], когда речь не идет о многоуровневой политике. Скрытый канал передачи информации через Интернет строится с помощью вписывания сообщения вместо последнего бита оцифрованного изображения, которое передается в качестве легального сообщения. Поскольку последний бит мало влияет на качество изображения, передача информации оказывается скрытой от субъекта ведущего перехват и допускающего передачу только легальных изображений. Хорошо известен метод борьбы с данным методом стеганографии, заключающийся в изменении формата изображения, например, с помощью компрессии. Данный метод уничтожает скрытый канал указанного вида.

Еще одним примером скрытого канала в аналогичной задаче является скрытый канал в TCP/IP протоколе, рассмотренный в работе Роуланда [21]. Поле ISN в TCP протоколе служит для организации связи клиента с удаленным сервером. Размер этого поля 32 бита. Используя это поле в 5 пакетах, было скрытно передано слово Hello.

Ряд скрытых каналов по времени, порожденных работой процессора, приведен в работе [19]. Особо следует выделить два примера каналов по времени, использующих возможности изменять длительности занятости в работе центрального процессора. В первом примере отправитель информации меняет время занятости CPU в течение каждого фрагмента времени, выделенного для его работы. Например, для передачи 0 и 1 одна длина промежутка времени кодирует 1, а другая - 0. В другом случае отправитель использует промежутки времени между обращениями к процессору. Более подробно об этих каналах можно прочитать в работе Хаскампа [14].

5. Модели скрытых каналов и их анализ

Модели скрытых каналов используются для разработки методов выявления скрытых каналов или, наоборот, для обоснования невозможности выявить подобные каналы. Традиционный метод выявления скрытых каналов [13] опирается на модель зависимости. Как определялось выше зависимости представляют из себя тройки (T, $\{S\}$, G), в которых изменение параметра T определяется изменением исходных параметров $\{S\}$, когда логическое выражение G принимает значение истина.

Пусть в рассмотренном ранее примере скрытого канала при использовании однонаправленного канала RS 232 условие G принимает значение истина, когда при передаче появляется фиксированный байт. В этом случае S есть время задержки выставления сигнала о возможности приема следующего байта. Агент нижнего уровня измеряет время задержки выставления сигнала на таймере T только тогда, когда передан байт, обращающий логическое выражение G в истину. Поиск данного скрытого канала наблюдателем за временами задержки выставления сигнала значительно сложнее, чем в приведенном ранее примере. Однако статистическими методами сам факт такой передачи можно распознать.

С методом зависимостей тесно связан метод поиска скрытых каналов на основе матрицы разделяемых ресурсов (SRM, Кемерер, 1980 г. [15]). В этом методе предполагается, что система полностью описывается переменными *a, b, c, d, ...* Анализ операций OP_1 проводится в матрице следующим образом. Строчки матрицы соответствуют атрибутам разделяемых ресурсов (в нашем примере *a, b, c,*

d, ...). Столбцы матрицы соответствуют операциям системы (OP_1 в нашем примере). Значения в клетках матрицы соответствуют действиям оператора над соответствующим атрибутом. Тогда матрица в нашем примере примет вид.

Разделяемый ресурс	OP_1
a	write
b	read
c	read
d	read

Данная матрица показывает потенциальные информационные потоки между переменными. Для анализа таких матриц были созданы пакеты прикладных программ.

Следующий вопрос, который возникает в таких задачах, можно ли создать «невидимые» для контролирующего субъекта скрытые каналы. В работе А.Грушо [2] доказано, что если противник знает схему контроля в системе защиты, то при выполнении определенных условий возможно построение невидимого для системы защиты срытого канала управления программно-аппаратным агентом в компьютерной среде. При этом «невидимость» понимается в абсолютном смысле, то есть доказывается невозможность выявления такого канала любыми методами и средствами. Аналогично в работе А.Грушо [3] при условии знания противником системы защиты доказывается возможность построения «невидимого» скрытого канала при общении программно-аппаратных агентов в открытой среде между собой. Полученные в этой работе результаты также носят абсолютный характер, то есть доказывается, что скрытый канал невозможно выявить любыми методами и средствами.

При использовании метода зависимостей следует отметить следующее:

- Предложенный аппарат и некоторые его реализации (например, метод матриц разделяемых ресурсов (SRM), предложенный в [15] и др.) является конструктивным инструментом поиска скрытых каналов;
- Не все скрытые каналы могут быть описаны с помощью данного метода.

В частности, существуют скрытые статистические каналы (см. работу Е.Тимониной [7]), которые не выявляются с помощью метода выявления зависимостей. Например, если T и S связаны между собой статистической зависимостью, то метод зависимостей может не выявить детерминированную связь между этими параметрами. Пусть в рассмотренном ранее примере скрытого канала при однонаправленном интерфейсе RS 232 условие G по-прежнему соответствует появлению в передаче фиксированного байта. В случаях, когда G истина, агент на уровне High генерирует случайную величину с распределением P_1, если он хочет передать вниз единицу. Если агент хочет передать ноль или не передает сообщение вообще, то задержка выставления сигнала происходит в соответствии с распределением P_0. Агент нижнего уровня накапливает средние значения времен задержки только при передаче указанного выше байта. Идентификация единицы или нуля производится агентом нижнего уровня с помощью статистического критерия. При этом контролирующий субъект, наблюдающий за задержками выставляемого сигнала на верхнем уровне, не знает при каком байте происходит изменение распределения времени задержки. Поэтому возможности выявления данного канала контролирующим субъектом значительно ниже возможности агента нижнего уровня распознать передаваемый сигнал. В работе [12] приводится пример соотношения параметров в аналогичном канале, при которых выявление скрытого канала контролирующим субъектом практически невозможно.

Одной из важнейших задач анализа скрытых каналов является оценка их пропускной способности. Пропускная способность скрытых каналов вычисляется методами теории информации. Часто пропускная способность скрытых каналов измеряется отношением количества скрытно переданной информации к количеству информации в легальном маскирующем скрытую передачу процессе или контейнере. Например в обычном цифровом изображении, содержащем 2,5 мегабит информации, манипуляцией последнего бита безопасно можно передать только 100 бит скрытого сообщения. С другой стороны, скрытый канал потайного копирования интересующего файла на дискету обладает пропускной способностью, равной объему копируемого файла. Другие примеры таких вычислений можно найти в работах А.Грушо, Е.Тимониной [4] и Деннинг [10]. В силу того, что скрытые каналы обладают, как правило, небольшой пропускной способностью, может сложиться мнение, что они не представляют опасности. Часто в таких случаях устанавливается порог на пропускную способность, ниже которого канал считается не опасным. Однако не следует забывать, что оценки пропускной способности, носят асимптотический характер и подход, связанный с ограничением пропускной способности, может оказаться неэффективным в реальных приложениях. Например, в работе Московиц [18] показано, что канал может иметь асимптотически нулевую пропускную способность, однако через него вполне можно передать скрытое сообщение небольшой длины.

6. Борьба со скрытыми каналами

Перехват информации, передаваемой по скрытым каналам, представляет большую сложность. Кажется, что здесь возникают только технологические сложности, связанные с регистрацией и анализом быстро протекающих процессов в компьютерных системах. Вместе с тем в работе А.Грушо и Е.Тимониной [5] доказано, что возможно создание производителем закладок в аппаратных системах, которые могут общаться между собой «невидимо» для большинства средств защиты.

В случае использования методов стеганографии решение задачи выделения скрытых сообщений представляется более оптимистичным. Примером успешного выявления стеганографических вставок является использование скрытого канала в поле ISN протокола TCP, упоминавшегося выше.

Наиболее эффективным способом борьбы со скрытыми каналами является их уничтожение. Например, в приведенных выше примерах скрытых каналов спуска информации при использовании интерфейса RS 232 встраивание между уровнями High и Low устройства, транслирующего байты и рандомизирующего задержку выставления сигнала на верхнем уровне, видимую на нижнем уровне, позволяет полностью уничтожить любой детерминированный скрытый канал по времени и существенно испортить скрытый статистический канал. Аналогичные методы успешно используются для защиты от скрытых каналов при скрытой передаче информации через открытые системы. Детальный обзор этих методов можно найти в работе Петиколаса [20].

К тематике скрытых информационных потоков близко подходит проблема «скрытого влияния». Для пояснения проблемы остановимся на скрытых каналах, которые физически не существуют. Для объяснения этой парадоксальной ситуации рассмотрим двухуровневую систему, в которой уровень High соответствует аналитической подсистеме, вырабатывающей варианты некоторого решения. Уровень Low соответствует подсистеме сбора информации из открытых источников и открытых сетей. Пусть подсистемы Low и High связаны однонаправленным каналом от Low к High, который позволяет реплицировать на верхнем уровне собранные на нижнем уровне данные. Пусть объект Y принадлежит уровню High, а объект X принадлежит уровню Low. Пусть совместное распределение состояний (x, y) объектов Y и X не равно произведению индуцированных вероятностей $P_X(x)$ и $P_Y(y)$. Тогда, как было отмечено ранее, согласно математической теории связи, от X к Y существует информационный поток, который измеряется средней взаимной информацией $I(X, Y)$. Но тогда точно такой же информационный поток существует от Y к X, измеряемый той же величиной. В случае однонаправленного канала такая симметрия имеет очень простую интерпретацию. На нижнем уровне, реплицируя объект X на верхний уровень, известно, что на верхнем уровне существует объект Y в точности с тем же состоянием, что и X. Если какой-то объект Z связан с объектом Y на верхнем уровне, и известно, что существует объект с данным видом связи, то естественно возникает информационный поток между X и Z. Причем этот информационный поток также симметричен, что означает возможность получения некоторой информации об объекте верхнего уровня, не имея физического канала от Z к X. Для того, чтобы лучше понять суть таких каналов, представим себе, что на верхнем уровне секретное решение принимается с помощью некоторого известного алгоритма только на основании информации, реплицированной с нижнего уровня. Зная это противник на нижнем уровне может из той же информации с помощью того же алгоритма получить точно такое же решение. Для офицера информационной безопасности это значит, что секретная информация верхнего уровня стала известна на нижнем уровне, что можно интерпретировать как существование некоторого скрытого канала, передающего информацию с верхнего уровня на нижний и несанкционированно снижающего гриф информации. Сам факт существования этого канала определяется симметрией информационного потока от объектов нижнего уровня на верхний. В общем случае те же рассуждения приводят нас к тому, что на нижнем уровне становятся известными некоторые вероятностные характеристики решения принимаемого на верхнем уровне. Таким образом, интерпретируется парадоксальный факт существования скрытого канала с верхнего уровня на нижний, хотя физически этот канал не существует. Из указанного примера следует два важных вывода. Первый вывод состоит в том, что многоуровневая политика не является гарантией безопасности, так как секретная информация верхнего уровня может стать известной на нижнем уровне независимо от способа реализации многоуровневой политики. Второй вывод состоит в том, что наиболее общая вероятностная трактовка информационного потока не позволяет просто разделить множество информационных потоков на разрешенные и не разрешенные.

Литература

1. Грушо А.А., Тимонина Е.Е. Теоретические основы защиты информации, М.: Агентство «Яхтсмен», 1996, 186 с
2. Грушо А.А. Скрытые каналы и безопасность информации в компьютерных системах // Дискретная математика, т.10, вып. 1, 1998.
3. Грушо А.А. О существовании скрытых каналов// Дискретная математика, т.11, вып. 1, 1999.
4. Грушо А.А., Тимонина Е.Е. Двойственность многоуровневой политики безопасности //Тезисы докладов конференции «Методы и технические средства обеспечения безопасности информации», С.-Петербург, изд-во СПбГТУ, 2000.
5. Грушо А.А., Тимонина Е.Е. Модель невлияния для сети// Тезисы докладов пятой международной Петрозаводской конференции «Вероятностные методы в дискретной математике», Обозрение прикладной и промышленной математики, научное изд-во «ТВП», Москва, 2000.
6. Кудрявцев В.Б., Алешин С.В., Подколзин А.С. Введение в теорию автоматов, М.: Наука, 1985.
7. Тимонина Е.Е. Механизмы контроля скрытых каналов / Труды международной конференции «Информационные технологии в науке, образовании, телекоммуникации, бизнесе», Украина, Крым, 20-30 мая 2002, стр. 152-153.
8. Шеннон К. Работы по теории информации и кибернетике, М.: Иностранная литература, 1963.
9. Biba K.J. Integrity Considerations for Secure Computer Systems //The MITRE Corp., Report No. MTR-3153 Revision 1, Electronic System Division, U.S. Air Force Systems Command, Technical Report Esd-TR-76-372, Bedford, Massachusetts, April 1997- 5.
10. Denning D.E. A Lattice Model of Secure Information Flow// Communications of ASM, 19:5, pp. 236-243, May 1976.
11. Department of Defense Trusted Computer System Evaluation Criteria, DoD, 1985.
12. Grusho A.A. Mathematical Models of the Covert Channels// Information Assurance in Computer Networks. Methods, Models, and Architectures for Network Security: International Workshop MMM-ACNS 2001 St. Petersburg, Russia, May 212-23, 2001/ Springer, 2001.
13. Handbook for the Computer Security Certification of Trusted Systems //NRL Technical Memorandum 5540:062A, 12 Feb. 1996.
14. Huskamp J.C. Covert Communications Channels in Timesharing Systems/ Technical Report UCB-CS-78-02, Ph.D. Thesis, University of California, Berkley, California, 1978.
15. Kemmerer R.A. Shared Resource Matrix Methodology: An Approach to Identifying Storage and Timing Channels //ACM Transactions on Computer Systems, 1:3, pp. 256-277, August 1983.
16. Lampson B.W. A Note of the Confinement Problem //Communications of ACM, 16:10, pp. 613-615, October 1973.
17. Millen J.K. Security Kernel Validation in Practice/ Communications of ASM, 19:5, May 1976.
18. Moscowitz I.S., Kang M.H. Covert Channels - Here to Stay?// Information Technology Division Naval Research Laboratory, Washington, DC 20375, 1995.
19. National Computer Security Center. A Guide to Understanding Covert Channel Analysis of Trusted Systems, 1993, NCSC-TG-030, ver. 1.
20. Petitcolas F. A.P., Anderson R.J., Kuhn M.G. Attacks on Copyright Marking Systems/ Second workshop on information hiding, in vol. 1525 of Lecture Notes in Computer Science, Portland, Oregon, USA, 14-17 April, 1998, pp. 218-238.
21. Rowland C.H. Covert Channels in the TCP/IP Protocol Suite// 11-14-1996, Psionic Technologies Inc., 2002.
22. Schneier B. Applied Cryptography: Protocols, Algorithms, and Source Code in C, John Wiley & Sons, New York, 2nd edition, 1996.
23. Tsai C.-R., Gligor V.D., Chandersekaran C.S. A Formal Method for the Identification of Covert Storage Channels in Source Code// IEEE Transactions on Software Engineering, v.16:6, June 1990, pp. 569-580.

О скрытых каналах и не только

Алексей Галатенко

Прежде чем ввести вас в курс дела, я хочу, чтобы вы поклялись в том, что все останется между нами.
Р. Стаут. «Если бы смерть спала»

Оправдания

Не имея личного опыта по части того, что происходит в приютах похоти, я не мог считаться экспертом в этой области.
Р. Стаут. «Слишком много клиентов»

Автор не является экспертом в области скрытых каналов и прекрасно осознает это. Данную статью он рассматривает как еще одну попытку убедить себя и, если повезет, читателя в продуктивности подхода к проблемам информационной безопасности с позиций технологии программирования, в необходимости рассматривать программно-технический уровень безопасности в общем контексте информационных технологий.

СОДЕРЖАНИЕ

Оправдания....12

К истокам....12

Коротко о традиционном подходе к проблеме скрытых каналов....15

Скрытые каналы и повседневная практика....17

Скрытые каналы и Java-аплеты....18

Заключение....19

Благодарности

Литература....20

К истокам

Чем выше ум, тем тень длиннее ляжет,
Отброшенная им на дальний мир.
Р. Браунинг. «Парацельс»

В статьях и обзорах, посвященных скрытым каналам, обычно пишут, что этот термин введен в работе Лэмпсона [1]. Важно отметить, однако, что Батлер Лэмпсон упоминает о скрытых каналах как бы

между делом, не они являются предметом исследования. Его трехстраничная заметка называется «О проблеме ограничения» («A Note on the Confinement Problem») и посвящена она, выражаясь современным языком, контролируемому выполнению (недоверенных) программ с целью помешать им организовать утечку конфиденциальной информации.

Идея дополнительного ограничения действий, которые разрешается выполнять программе, (помимо применения имеющихся в операционной системе механизмов контроля доступа), является исключительно важной и глубокой, опередившей свое время. По сути Лэмпсон пунктиром обозначил будущие модели безопасности для Java-аплетов — от "песочницы" до контроля по стеку вызовов, причем сделал это на 25 лет раньше разработчиков Java (заметка была сдана в редакцию в июле 1972 года, а по ссылкам видно, что он занимался проблемами динамической защиты еще в 1960-е годы).

Лэмпсон рассматривает следующую задачу. Пусть клиент вызывает некоторый сервис, передавая ему в качестве параметров конфиденциальную информацию, утечка которой нежелательна. Спрашивается, как следует ограничить поведение произвольного сервиса? (В 1972 году сервис представлял собой процедуру, вызываемую из клиентской программы; о распределенных архитектурах речь не шла, но многопроцессные конфигурации были обычным явлением.)

Подчеркнем, что речь идет о создании «песочницы» для произвольной программы. Если ограничения окажутся нарушенными, выполнение сервиса должно аварийно завершаться.

Чтобы понять характер налагаемых ограничений, Лэмпсон сначала исследует возможные каналы утечки информации, выделяя следующие:

- Если у сервиса есть память, он может сохранить клиентскую информацию, дождаться, когда его вызовет хозяин, и передать тому сохраненную информацию;
- Сервис может записать информацию в постоянный файл в каталоге хозяина;
- Сервис может создать временный файл (что само по себе вполне законно: как же без временных файлов?); хозяин может периодически проверять его (файла) существование и прочитать информацию прежде, чем сервис завершит работу и удалит все временное;
- Сервис может послать сообщение процессу, контролируемому хозяином;
- Сервис может закодировать информацию в счете за свои услуги, поскольку хозяин должен получить копию этого счета; ограничения на формат счета способны свести объем утечки к нескольким десяткам бит, но полностью ликвидировать ее нереально;
- Если могут возникать конфликты по ресурсам, сам факт конфликта может быть использован для передачи информации (чуть ниже мы подробнее рассмотрим этот канал утечки);
- Сервис может варьировать отношение вычислительной активности к темпу подкачки страниц или числу операций ввода/вывода, кодируя таким способом информацию; параллельно работающий процесс способен наблюдать поведение системы и получать передаваемую информацию; это зашумленный канал, его пропускная способность невелика, но он есть, и им можно воспользоваться.

Всегда интересно не просто теоретически знать, что каналы утечки существуют, но и понять на практике, как они могут быть устроены. Лэмпсон описывает «эксплойт» для использования конфликтов по ресурсам (мы приведем его в несколько модифицированном виде). Пусть, например, один и тот же файл нельзя параллельно открыть из двух процессов (при попытке сделать это фиксируется ошибка). Данный факт можно использовать для побитной передачи информации. Две следующие процедуры, написанные на АЛГОЛоподобном языке обеспечивают, соответственно, установку нужного бита и его проверку.

```
procedure setbit (file, bit);
   Boolean bit;
begin
   if bit = true then
      loop1: open (file, loop1)
   else
      close (file)
end;

Boolean procedure testbit (file);
begin
   testbit : = true;
   open (file, loop2);
   testbit := false;
   close (file)
loop2: end;
```

(К своему стыду, автор не знает, как окружение АЛГОЛ-программ реагирует на попытку закрыть неоткрытый файл).

Сейчас далеко не все помнят тонкости АЛГОЛа-60, поэтому нужно пояснить по крайней мере два момента. Во-первых, метки в АЛГОЛе являются

допустимым типом данных, значения которого можно передавать в качестве параметров. Во-вторых, если (занятый) файл открыть не удалось, управление нелокально передается на метку, заданную в качестве второго аргумента процедуры open. Кстати, это более практичный способ обработки исключительных ситуаций, чем проверка кодов возврата, которую зачастую забывают сделать, что является одним из источников уязвимостей программных систем.

Теперь, располагая элементарными операциями, можно организовать передачу бита между процессами. Для этого Лэмпсон использует три файла: data, sendclock и receiveclock.

В программе-сервисе может присутствовать такой фрагмент (написанный нами с некоторыми вольностями):

```
sendingabit:
   setbit (data, bitbeingsent);
   while testbit (receiveclock) = false do
      ;
   setbit (sendclock, false);
   while testbit (receiveclock) = true do
      ;
```

Процесс-получатель может осуществлять прием бита так:

```
receiver:
   while testbit (sendclock) = false do
      ;
   receivedbit : = testbit (data);
   setbit (receiveclock, true);
   while testbit (sendclock) = true do
      ;
   setbit (receiveclock, false);
```

Переходя к правилам ограничения (контролируемого выполнения), Лэмпсон прежде всего указывает, что ограничиваемая программа не должна иметь возможности сохранять информацию между вызовами. Для процедур это условие легко проверяется: оно означает отсутствие обращений к нелокальным переменным.

Если нелокальная память отсутствует, то для успешной реализации ограничения достаточно, чтобы ограничиваемая программа не вызывала других программ. Это правило полной изоляции по сути совпадает с моделью "песочницы" в версии JDK 1.0. Разумеется, Лэмпсон тут же отвергает его как нереалистичное, поскольку, как показывают два последних из перечисленных выше примеров утечек, в числе прочих должны быть запрещены явные и неявные вызовы супервизора (ядра ОС). Попытка применить правило полной изоляции «наскозь» и потребовать, чтобы ограничивались все вызываемые программы, не проходит, поскольку супервизор нельзя ограничивать.

Чтобы исправить ситуацию, предлагается, как и следовало ожидать, поделить всех на чистых и нечистых, то есть на тех, кому доверяют, и тех, кого ограничивают. В результате получается следующее правило: если ограничиваемая программа вызывает недоверенную, то последнюю также необходимо ограничивать.

Дело осталось за малым — написать доверенный супервизор. Как показывают два последних из перечисленных выше примеров утечек, дело это непростое, поскольку используемые для этого каналы могут быть самыми неожиданными. Тем не менее Лэмпсон придерживается оптимистической точки зрения: каналов утечки, конечно, на удивление много, но все-таки конечное число. Необходимо их перенумеровать, а потом и заблокировать. В качестве отправной точки предлагается следующая классификация каналов:

- Разного рода память, управляемая супервизором, в которую может писать ограничиваемая программа (в рассматриваемых примерах — сервис), а читать — неограничиваемая (вскоре после записи или позднее); все приведенные выше примеры, кроме двух, относятся к этому классу;
- Легальные каналы, используемые ограничиваемой программой (например, счет за обслуживания);
- Скрытые каналы, вообще не предназначенные для передачи информации, (например, влияние сервиса на загруженность системы).

Вот в каком контексте вводится понятие скрытого канала (covert channel) и как оно определяется Лэмпсоном. Обратим внимание читателя на то, что все каналы по памяти отнесены к другому классу утечек. Важно и то, что помимо скрытых, Лэмпсон рассматривает и так называемые «подсознательные» или потайные каналы (легальные каналы, по которым передаются данные, допускающие нестандартную интерпретацию и, следовательно, которые могут служить каналом утечки конфиденциальной информации), хотя используемый в настоящее время термин subliminal channel, по-видимому, введен Б. Шнейером [2] примерно двадцать лет спустя.

По мнению Лэмпсона предотвратить утечки через память довольно просто. Например, чтобы избавиться от блокировок при совместном доступе к ресурсам, можно при попытке записи копировать файл и бесконфликтно предоставлять копию для чтения ограничиваемой программой. (Заметим, что идея эта весьма глубока, только лучше при попытке

записи открывать новую версию файла и писать в нее, применяя в дальнейшем механизмы конфигурационного управления.)

Для блокирования легальных и скрытых каналов предлагается следующий принцип маскирования: ограничиваемая программа должна позволять вызывающему полностью определять вывод в легальные и скрытые каналы. Таким образом каналы маскируются вызывающим (клиентом).

Это — центральная идея работы Лэмпсона. Программа ограничивается в соответствии с ожидаемой семантикой и специфическими потребностями вызывающего. Например программа для просмотра документов не должна изменять или создавать файлы. Напротив компилятору без создания файлов (временных и постоянных) не обойтись.

Вообще говоря при разных вызовах одного сервиса ограничения могут быть разными. Тем более они могут быть разными для разных сервисов. Если сервис не в состоянии удовлетворить требования клиента, вызов отвергается.

Для скрытых каналов Лэмпсон формулирует еще одно правило — проведения в жизнь: супервизор обязан обеспечить соответствие вывода ограничиваемой программы в скрытые каналы спецификациям вызывающего. Считается, что концептуально это несложно: можно искусственно замедлять выполнение программы, генерировать фиктивные обращения к дискам и т.п., короче говоря, зашумлять скрытые каналы. Цена проведения в жизнь может быть большой. Более практичной альтернативой (если клиент готов допустить некоторую утечку) является ограничение пропускной способности скрытых каналов.

Конечно, можно отметить, что некоторые положения работы Лэмпсона, касающиеся концептуальной простоты или принципиальной реализуемости выявления и блокирования каналов утечки (в частности, скрытых каналов), остались необоснованными. Более того, как показали последующие исследования, они неверны. Тем не менее эта работа продолжает оставаться актуальной и в наше время, а ее идейный потенциал далеко не исчерпан. Тематика ограничения (контролируемого выполнения) программ ждет дальнейшего развития.

Коротко о традиционном подходе к проблеме скрытых каналов

Прими с достоинством то, что не можешь изменить.
Сенека

Лэмпсон понимал, что без учета семантики программ невозможно управлять доступом и, в частности, не допускать утечек информации. К огромному сожалению, эта идея не была воспринята. Последующие усилия в основном оказались направленными на разработку и реализацию универсальных и формальных, но недостаточно содержательных моделей безопасности (в особенности моделей доступа), а также на выявление и оценку пропускной способности скрытых каналов для таких моделей. Более содержательная и важная для практики задача ограничения программ по сути оказалась забытой.

Традиционным стало следующее определение скрытого канала (см. [3]). Пусть дана модель недискреционной политики безопасности M и ее реализация I(M) в операционной системе. Потенциальная связь между субъектами $I(S_i)$ и $I(S_j)$ в I(M) называется скрытым каналом тогда и только тогда, когда эта связь между S_i и S_j в модели M не разрешена.

Несмотря на формальный стиль и пугающее слово «недискреционной», смысл этого определения довольно прост. С одной стороны, по сравнению с работой Лэмпсона, понятие скрытого канала стало трактоваться шире. Фактически к нему отнесены все виды передачи информации, нарушающие политику безопасности. С другой стороны, наложено ограничение на дисциплину разграничения доступа, реализуемую в ОС. Она не должна сводиться к произвольному (дискреционному, согласно официальной терминологии) управлению доступом. Это значит, что круг рассматриваемых систем сужен до весьма немногочисленных, хотя и критически важных, режимных конфигураций, использующих многоуровневую политику безопасности и принудительное (мандатное) управление доступом.

Напомним, что в «Оранжевой книге» требования, касающиеся скрытых каналов, фигурируют, начиная с класса B2, а мандатный доступ появляется в классе B1. В тех странах, где нормативные документы в области информационной безопасности построены на основе «Оранжевой книги», режимные организации должны бороться со скрытыми каналами и по содержательным, и по формальным причинам, для них и реализация принудительного управления доступом, и блокирование скрытых каналов — действия необходимые. В то же время, они весьма

сложны. Например, в упомянутой выше работе [3] обосновывается необходимость анализа исходного текста (наряду с анализом спецификаций) ядра ОС для выявления скрытых каналов по памяти и утверждается, что на ручной анализ ядра Secure Xenix требуется два человеко-года. Разумеется, анализ можно автоматизировать, что и было сделано, после чего было выявлено 25 переменных, потенциально пригодных для организации скрытых каналов. Любопытно отметить, что существование пяти каналов стало возможным из-за конфликтов между программным интерфейсом ядра Secure Xenix и правилами мандатного доступа; очевидно, без потери совместимости ликвидировать эти каналы невозможно.

В подавляющем большинстве случаев многоуровневая политика безопасности направлена на сохранение конфиденциальности. С этой целью запрещаются информационные потоки «вниз», с верхних уровней секретности (доверия) на нижние. В таких условиях передатчиком в скрытом канале должна служить "троянская" программа с высоким уровнем доверия.

Скрытые каналы по памяти были детально рассмотрены в предыдущем разделе. В принципе, скрытые каналы по времени устроены аналогично примеру с конфликтами при открытии файлов; различия касаются в основном способов кодирования передаваемой информации.

Чтобы организовать скрытый канал по времени, нужен разделяемый ресурс (например, процессор) с возможностью воздействия со стороны одного процесса и выявления этого воздействия со стороны другого, а также двое часов — эталонные и сигнальные. Часы могут быть виртуальными и реализовываться в виде последовательности наблюдаемых событий. Передача информации осуществляется за счет модуляции последовательности сигнальных событий, прием — путем измерения модуляции относительно эталонных часов.

В работе [4] рассматривается построение скрытого канала по времени, использующего модуляцию занятости процессора. Идея состоит в следующем. На верхнем уровне доверия действует одна программа-передатчик. За один такт эталонных часов передатчик выполняет два вида циклов. Сначала (фаза 1) он почти не тратит процессорное время, сразу отдавая управление планировщику; затем (фаза 2), наоборот, занимает процессор максимально плотно. Приемник пытается определить относительную длительность этих двух фаз. Для этого приемник реализуется в виде двух процессов. Один подсчитывает, сколько раз он получил доступ к процессору, в цикле увеличивая (разделяемый) счетчик и тут же отдавая управление планировщику; другой в конце такта часов считывает и сохраняет значение счетчика, обнуляет его и откладывается до конца следующего такта.

Со скрытыми каналами можно бороться двумя способами: пытаться блокировать их полностью или уменьшать пропускную способность.

Для полного блокирования скрытого канала нужно устранить разделение ресурсов, видимое для процессов с низким уровнем доверия; последние должны выполняться так, будто кроме них в системе никого и ничего нет. Подобного невлияния можно добиться либо избегая конфликтов (например, за счет устранения разделения ресурсов), либо незаметным образом разрешая их в пользу «низших». Это, в свою очередь, порождает проблему, аналогичную инвертированию приоритетов: процесс с высоким уровнем доверия рискует получить слишком мало ресурсов, а совершаемые им транзакции могут слишком часто откладываться. Для борьбы с дискриминацией «секретных» процессов приходится идти на новые хитрости (см., например, [5]).

Разумеется, в сколько-нибудь сложной системе избежать видимого разделения ресурсов невозможно. Идея реализации доменов безопасности с непроницаемыми границами и аппаратно односторонними междоменными каналами (см. [6]) относится скорее к области курьезов, поскольку игнорирует реально используемые коммуникационные протоколы. Следовательно, с существованием некоторых скрытых каналов приходится мириться, пытаясь, однако, уменьшить (или, по крайней мере, оценить) их пропускную способность.

Если считать, что такт T эталонных часов постоянен, и за один такт можно различить N модуляций, то пропускная способность скрытого канала по времени оценивается величиной log2(N)/T. Для уменьшения пропускной способности можно «сбивать» часы (эталонные и/или сигнальные), «зашумлять» канал еще каким-либо способом (например, обслуживая фиктивные запросы), уменьшать разрешающую способность часов (увеличивая такт). (Заметим, что предпочтительнее воздействовать на эталонные, а не сигнальные часы, так как первое дает линейный эффект, а второе — лишь логарифмический). Любопытно отметить, что в локальной сети корпорации Боинг, сертифицированной по классу A1, разрешающая способность часов, доступных недоверенным процессам, составляет одну секунду (см. [4]). Как говорится, страшнее скрытого канала зверя нет, и для борьбы с ним приходится применять весьма сильнодействующие средства, существенно снижающие эффективность и осложняющие работу приложений. Впрочем, как показывает рассмотренный выше пример, можно добраться до более тонких внутренних часов планировщика; он демонстрирует также, насколько

сложны анализ пропускной способности скрытых каналов и борьба с ними.

Первопричина существования скрытых каналов и невозможность их устранения при традиционном подходе к построению информационных систем видится нам в следующем. Поскольку такие формальные модели безопасности, как известная модель Белла-Лападула, разграничивают доступ «в принципе», но не содержат понятия времени и не регламентируют конкуренцию за ресурсы, то есть ситуации, когда «в принципе ресурс использовать можно, только в данный момент нельзя — он занят», при любом распределении прав доступа разного рода сигнальные события и, в частности, коллизии вследствие конкуренции могут быть использованы для организации скрытых каналов.

С потайными каналами (см. выше раздел 1) дело обстоит еще хуже. Представляется очевидным, что если не формализовать структуру данных, передаваемых программами по легальным каналам (Лэмпсон указывал на необходимость подобной формализации), последние всегда можно использовать для скрытой передачи информации. Строгое доказательство невозможности устранения и даже обнаружения потайных каналов при весьма общих предположениях относительно схем контроля имеется в работах А.А. Грушо [7], [8].

Перечисленными причинами объясняется необходимость развития отдельной науки под названием «анализ скрытых каналов», а также принципиальная невозможность полностью блокировать утечку конфиденциальной информации универсальными средствами, не учитывающими семантику программ.

Скрытые каналы и повседневная практика

«Баржа с грузом трески вошла в порт. Поспешите, товар скоропортящийся.»
Г. Мопассан. «Заведение Телье»

В повседневной практике скрытые каналы трактуются шире, чем в теории. Расширение происходит по четырем направлениям:

- Рассматриваются системы с произвольной дисциплиной управления доступом (а не только с многоуровневой политикой безопасности);
- Рассматриваются не только нестандартные каналы передачи информации, но и нестандартные способы передачи информации по легальным каналам (потайные каналы);
- Рассматриваются угрозы не только конфиденциальности, но и целостности;
- Рассматриваются не только однонаправленные, но и двунаправленные каналы.

В повседневной практике скрытые каналы чаще всего возникают из-за возможности доступа к данным несколькими способами. Например, если в корневом каталоге файловой системы располагается tftp-сервер, он позволит получить всем пользователям доступ на чтение ко всем файлам, независимо от установленных прав доступа. Второй пример: если есть программа со взведенным битом переустановки действующего идентификатора пользователя, владельцем root и ... уязвимостью, то обычный пользователь может проэксплуатировать уязвимость, захватить привилегии суперпользователя и нарушить конфиденциальность и целостность чего угодно. Третий пример: пароль в незашифрованном виде, хранящийся в оперативной памяти и сбрасываемый в файл с образом памяти при аварийном завершении. Очевидно, число подобных примеров можно умножать до бесконечности.

Нестандартные способы передачи информации по легальным каналам получили название «подсознательных» или потайных каналов (subliminal channels), хотя по аналогии с наложенными сетями их лучше было назвать наложенными скрытыми каналами. Потайные каналы используют тогда, когда имеется легальный коммуникационный канал, но политика безопасности запрещает передавать по нему определенную информацию; иными словами, информацию передавать можно, но она не должна выглядеть подозрительно (в соответствии с некими, обычно не очень четкими критериями). Помимо разведчиков, в потайных каналах нуждаются хозяева "троянских" программ, таких, например, как Back Orifice или Netbus; если канал взаимодействия с ними будет явным, «троянца» быстро вычислят и уничтожат. Подобные каналы, используемые для управления, должны быть двунаправленными.

Потайной канал возможен тогда, когда в передаваемых легальных данных есть незначащие или почти незначащие биты, например, некоторые биты в заголовках IP-пакетов или младшие биты интенсивности цветов в графическом файле, присоединенном к письму. (Электронная почта — иде-

альный легальный канал, на который удобно накладывать скрытые.) В работе [9] рассматривается другой пример: нестандартный алгоритм электронной подписи, оставляющий место для тайных сообщений, не препятствующих проверке аутентичности ЭЦП.

Для нарушения целостности чаще всего используют уязвимости, связанные с переполнением буферов. Если тем самым изменяется содержимое стека вызовов, то перед нами еще один пример скрытого канала, основанного на возможности доступа к данным нестандартным способом. Применяются также атаки пользовательских процессов на целостность транзакций, выполняемых процессами привилегированными (соответствующий англоязычный термин — race condition). Возможность подобной атаки появляется, если временные данные для транзакции располагаются в общедоступных каталогах, таких как /tmp.

Разумеется, при расширительном толковании понятия скрытого канала проблемы, описанные в предыдущем разделе, не только остаются, но и обостряются; кроме того, к ним добавляются новые. С этими проблемами можно бороться двумя способами: формальным и содержательным.

Формальный способ усиления безопасности состоит в попытке добавить к употребительным операционным системам, таким как Linux, возможности, реализующие требования классов B2 и старше из «Оранжевой книги», то есть реализовать в ядре ОС мандатный доступ ко всем ресурсам и провести анализ скрытых каналов. Вне зависимости от выполнимости последнего пункта, подобный путь представляется тупиковым. Одно дело анализировать скрытые каналы в операционной системе мэйнфрейма, изначально спроектированной с учетом требований безопасности и прошедшей многолетнюю апробацию (см., например, [10]) и совсем другое — в ОС, содержащей ошибки, связанные с переполнением буферов. Для подлинной безопасности нужно не только добавление принудительного управления доступом, но и существенное перепроектирование ОС.

Содержательный способ борьбы со скрытыми каналами состоит в выстраивании эшелонированной обороны; утечки информации признаются неизбежными, но их пытаются локализовать и отследить. Для иллюстрации рассмотрим следующий гипотетический пример. Банк, информационная система которого имеет соединение с Интернет, защищенное межсетевым экраном, приобрел за рубежом автоматизированную банковскую систему (АБС). Изучение регистрационной информации экрана показало, что время от времени за рубеж отправляются IP-пакеты, содержащие какие-то непонятные данные. Стали разбираться, куда же пакеты направляются, и оказалось, что идут они в фирму-разработчик АБС. Возникло подозрение, что в АБС встроены "троянская" программа и скрытый канал, чтобы получать информацию о деятельности банка. Связались с фирмой и в конце концов выяснили, что один из программистов не убрал из поставленного в банк варианта отладочную выдачу, которая была организована сетевым образом (как передача IP-пакетов специфического вида, с явно заданным IP-адресом рабочего места этого программиста). Если бы не межсетевой экран, канал оставался бы скрытым, а конфиденциальная информация о платежах свободно гуляла по сетям.

Конечно, мысль о необходимости эшелонированной обороны не нова; надо только не забыть реализовать ее на практике...

Еще один практически важный в данном контексте архитектурный принцип — разнесение доменов выполнения для приложений с разным уровнем доверия на разные узлы сети. Если такие приложения будут функционировать в распределенной среде клиент/сервер, ограничить их взаимное влияние (и, следовательно, заблокировать скрытые каналы) будет существенно проще, чем в случае единых многопользовательских систем.

Скрытые каналы и Java-аплеты

«Подразумевается, что Вульф приложит все силы для сохранения в тайне фактов, способных нанести ущерб корпорации; при несоблюдении этого условия он теряет право на гонорар.»
Р. Стаут. «Слишком много клиентов»

Одной из актуальных и практически важных проблем информационной безопасности является пресечение вредоносных действий со стороны мобильных агентов и, в частности, Java-аплетов. Подобные агенты — обязательная составляющая систем электронной коммерции, электронного голосо-

вания и других приложений, ориентированных на массовое, повседневное использование.

Универсальной модели безопасности, реализованной на платформе Java 2 и ориентированной на защиту локальных ресурсов от несанкционированного доступа со стороны аплета, недостаточно для того, чтобы предотвратить утечку конфиденциальной информации по скрытым каналам, поскольку такую информацию может передать аплету сам пользователь (в надежде на корректность поведения аплета). Хотелось бы, чтобы кроме надежды, у пользователя были и другие основания для доверия. Для этого необходимо ограничить поведение аплета в соответствии с его спецификациями, что означает возврат к истокам, к постановке задачи, предложенной Лэмпсоном тридцать лет назад.

В работе [11] рассматривается пример простого протокола из области электронной торговли, возникающие при этом сложности с обеспечением конфиденциальности и возможные пути их преодоления. Суть примера в следующем. Имеются три субъекта: покупатель, продавец и банк, обслуживающий покупателя. Для оформления заказа покупатель загружает с сервера продавца Java-аплет, осуществляющий ввод данных о заказываемом товаре и реквизитах счета покупателя. Аплет должен передать эту информацию продавцу, предварительно зашифровав банковские реквизиты имеющимся у покупателя ключом банка (продавцу знать реквизиты счета покупателя не полагается, ему нужно лишь получить от банка плату за покупку).

Очевидно, у аплета есть много способов разной степени скрытности передать продавцу конфиденциальную информацию о счете покупателя: изменить представление заказа зависящим от реквизитов образом, зашифровать реквизиты своим ключом (а не ключом банка) и т.п. Идея ограничения аплетов, предлагаемая в [11], состоит в том, чтобы вместе с интерпретируемым кодом поступали спецификация свойств конфиденциальности и допускающее автоматическую проверку доказательство того, что байт-код соответствует спецификации. В свою очередь, в спецификации задаются зависимости между изменением исходных данных (вводимых пользователем) и наблюдаемым поведением аплета. Эти зависимости должны быть в точности такими, как предписывает протокол. Например, результат шифрования реквизитов должен меняться при их изменении, равно как и при изменении банковского ключа; если последнее утверждение неверно, значит аплет использует для шифрования нелегальный ключ. Еще пример: данные о заказываемом товаре, передаваемые продавцу, не должны меняться при изменении реквизитов.

Мы не будем вдаваться в тонкости применяемого в [11] формализма. Отметим лишь, что предлагаемый подход представляется весьма перспективным; это подтверждается прототипной реализацией. Он не решает всех проблем; со скрытыми каналами по времени бороться по-прежнему трудно, хотя в принципе можно варьировать уровень детализации наблюдаемого поведения, добиваясь видимости все более тонких эффектов.

Заключение

«На этот раз я буду умнее», — подумал он, доставая камни из сумки и пряча их в карман. Это ведь портовый город, а в порту, как верно заметил тот, кто его обокрал, всегда полно жуликов.
П. Коэльо. «Алхимик»

Изучение проблематики скрытых каналов показывает, как важно правильно поставить задачу и рассматривать ее не изолированно, а в реальном окружении.

На наш взгляд, правильная постановка, связанная с контролируемым выполнением (ограничением) программ, с самого начала предложенная Лэмпсоном, в дальнейшем незаслуженно отошла на второй план. Попытки бороться со скрытыми (и потайными) каналами с помощью формальных методов, не учитывающих семантику программ, как показывают результаты А.А. Грушо, обречены на неудачу.

В реальном окружении проблема утечки конфиденциальной информации стоит особенно остро для распределенных систем с мобильными агентами. В настоящее время существуют перспективные подходы к ее решению.

На практике требуется блокировать скрытые каналы всех видов: как те, что угрожают конфиденциальности, так и представляющие угрозу целостности программ и данных. В частности, необходимо выявлять и пресекать общение "троянских" программ со своими хозяевами (равно как и попытки внедрения таких программ).

Следование принципам архитектурной безопасности (эшелонированность обороны, разнесение доменов выполнения) помогает бороться как со скрытыми каналами, так и с последствиями их функционирования. При современной технологии создания информационных систем выявить и блокировать все скрытые каналы невозможно. Нужно заставить злоумышленников выстраивать как можно более длинные цепочки из таких каналов; обнаружить и разорвать подобную цепочку существенно проще, чем отдельный канал.

Благодарности

Здесь аванс в десять тысяч долларов. Если бы я дал вам чек, об этом могли бы разнюхать те, от кого я хочу сохранить это в тайне. Расписка не нужна.
Р. Стаут. «Если бы смерть спала»

Автор признателен И.А. Трифаленкову, которому принадлежит общий замысел данной статьи, определивший ее практическую направленность.

Литература

Теперь вы знаете обо мне самое худшее, если не считать моим самым дурным поступком стихотворение «Реквием грызуну». Оно было напечатано в школьной газете.
Р. Стаут. «Если бы смерть спала»

1. Lampson B.W. A Note of the Confinement Problem. — Communications of ACM, v. 16, n. 10, Oct. 1973, pp. 613-615.
2. Schneier B. Applied Cryptography: Protocols, Algorithms, and Source Code in C. — John Wiley & Sons, New York, 2nd edition, 1996.
3. Tsai C.-R., Gligor V.D., Chandersekaran C.S. A Formal Method for the Identification of Covert Storage Channels in Source Code. — IEEE Transactions on Software Engineering, v. 16, n. 6, Jun. 1990, pp. 569-580.
4. James J.V.A, Darby D.B., Schnachenberg D.D. Building Higher Resolution Synthetic Clocks for Signalling in Covert Timing Channels. — Proceedings of the Eight IEEE Computer Security FOundations Workshop (CSFW'95). — IEEE, 1995, pp. 85-95.
5. Son S.H., Mukkamala R., David R. Integrating Security and Real-Time Requirements Using Covert Channel Capacity. — IEEE Transactions on Knowledge and Data Engineering, v. 12, n. 6, Nov.-Dec. 2000, pp. 865-879.
6. Davidson J.A. Asymmetric Isolation. — Proceedings of the 12th Annual Computer Security Applications Conference (ACSAC). — IEEE, 1996, pp. 44-54.
7. Грушо А.А. Скрытые каналы и безопасность информации в компьютерных системах. — Дискретная математика, т. 10, вып. 1, 1998.
8. Грушо А.А. О существовании скрытых каналов. — Дискретная математика, т. 11, вып. 1, 1999.
9. Jan J.-K., Tseng Y.-M. New Digital Signature with Subliminal Channels on the Discrete Logarithm Problem. — Proceedings of the 1999 International Workshops on Parallel Processing. — IEEE, 1999, pp. 198-203.
10. Симонов С., Колдышев П. Обеспечение информационной безопасности в вычислительных комплексах на базе мэйнфреймов. — Jet Info, 2002, 4.
11. Dam M., Giambiagi P. Confidentiality for Mobile Code: The Case of a Simple Payment Protocol. — Proceedings of the 13th IEEE Computer Security FOundations Workshop (CSFW'00). — IEEE, 2000, pp. 233-245.

Jet Info
ИНФОРМАЦИОННЫЙ БЮЛЛЕТЕНЬ
Издается с 1995 года
Издатель: компания Джет Инфо Паблишер

Главный редактор: Дмитриев В.Ю. *(vlad@jet.msk.su)*
Россия, 127006, Москва, Краснопролетарская, 6
тел. (095) 972 11 82, 972 13 32
факс (095) 972 07 91
email: JetInfo@jet.msk.su
http://www.jetinfo.ru

Подписной индекс по каталогу Роспечати 32555