Скрытые коммуникации и скрытые каналы

Скрытые коммуникации и скрытые каналы

Скрытый канал (covert channel) — это коммуникационный канал, пересылающий информацию методом, который изначально для этого не был предназначен.

Персонажи детективных и шпионских романов пользуются скрытыми каналами постоянно – журнал «Огонек» в левой руке, сообщающий, что операция отменяется, цветок на подоконнике, предупреждающий профессора Плейшнера об опасности, количество сальто, исполненное цирковым артистом для передачи информации о количестве танков на полигоне, – все это примеры скрытых каналов.

Техника скрытых каналов основывается на том простом факте, что любое изменение состояния какого-то объекта несет информацию. И обнаружить закономерность в изменениях состояния объекта, которое изначально никак не было предназначено для передачи информации, бывает очень сложно. За высокую степень скрытности многие каналы такого типа расплачиваются низкой скоростью передачи информации. Например, наличие или отсутствие цветка на подоконнике несет в себе 1 бит информации. Количество сальто в цирковом номере более информативно, но и с помощью такого способа много данных не передашь.

Теперь рассмотрим несколько примеров из практики информационных технологий. Пусть, например, имеются два процесса: процесс А, который имеет высокий уровень допуска, и процесс В, имеющий низкий уровень допуска. Злоумышленник имеет возможность получать информацию *только* от процесса В. В соответствии с мандатным способом управления процесс В не имеет легальной возможности прочитать некие данные с высоким уровнем секретности. Но к ним имеет доступ процесс А, который однако в соответствии с тем же правилом мандатного способом доступа не имеет права передавать данные на нижележащий уровень процессу В. Возникает вопрос, существует ли для процесса А какой-нибудь обходной, не вызывающий подозрений способ передать секретные данные процессу В?

Ответ – да. Таким способом может быть, например, варьирование степенью заполнения буфера при выводе легальных данных. Пусть процесс А, выполняя легальную операцию вывода, изменяет объем выводимых данных в соответствии с каким-либо условным кодом, известным процессу-адресату В. И пусть процесс В имеет возможность анализировать состояние буфера. Таким образом процесс А, выполняя казалось бы вполне законные действия, может передать процессу В в закодированном виде секретные данные. Подчеркнем, что эти данные, не имеют ничего общего с легально выводимой информацией. Здесь можно провести аналогию с передачей информации модулированным синусоидальным сигналом, когда информация кодируется с помощью варьирования амплитуды. Именно такая идея положена в основу скрытого канала.

Скрытый канал – это механизм для передачи информации, не предусмотренный разработчиком информационной системы. Естественно, что передача данных по скрытому каналу не контролируется обычными механизмами безопасности ОС, такими как аутентификация и авторизация, именно поэтому наличие скрытых каналов очень опасно. Недаром все стандарты безопасности информационных систем, такие как «Оранжевая книга» и «Общие критерии», уделяют скрытым каналам большое внимание и требует анализа системы на наличие таких каналов при сертификации.

Кроме скрытых каналов, существуют также ***скрытые коммуникации***. Скрытые коммуникации используют для передачи сообщений *легальные* каналы, однако действуют таким образом, что эти сообщения незаметны для легальных пользователей, так как они скрыты внутри других сообщений, используемых как контейнеры. Наиболее популярным примером скрытых коммуникаций является помещение секретного сообщения в биты цифровой фотографии, внешний вид которой от такой операции практически не отличается от исходного. Скрытые коммуникации также называют subliminal channel, что дословно переводится как *подсознательный канал*, канал, находящийся ниже уровня восприятия; есть также предложение называть такие каналы *потайными*[1].

Общей особенностью скрытых каналов и скрытых коммуникаций является то, что здесь скрывается не только содержание сообщения, но и сам *факт* коммуникации. Насколько сам факт коммуникации может оказаться ценным для разведки говорит программа PRIZM, о наличии которой мир узнал из утечек Эдварда Сноудена. Эта программа собирает *метаданные* об электронных коммуникациях, то есть данные о том, кто с кем, где и когда переписывался. И хотя собственно содержание сообщений не включается в собираемую информацию, метаданные сами по себе могут помочь раскрыть заговор или выявить агентов. Да и в личной жизни просто факт переписки одного из супругов с третьим лицом может привести к неожиданным последствиям.

Скрытыми коммуникациями занимается *стеганография*, поэтому такой способ передачи секретной информации иногда называется *стегоканалом*. Стеганография, как и криптография, имеет древнюю историю, и человечество придумало достаточно много остроумных способов помещения секретных данных в невинные с виду сообщения. В информационных системах не всегда можно провести четкую грань между техникой скрытых каналов и скрытых коммуникаций, мы увидим это немного дальше.

Примеры скрытых каналов

Существуют два типа скрытых каналов: ***скрытый канал памяти*** (*covert storage channel*) и ***скрытый временной канал*** (*covert timing channel*). В первом случае для скрытной передачи информации используется модулирование характеристик памяти, таких как адреса, объем свободной памяти и др. Процесс использует скрытый временной канал, если для кодирования информации он варьирует временные характеристики, связанные с его собственным выполнением, например, в мультипрограммном режиме, это может быть использованная доля кванта процессорного времени.

Большое количество работ по анализу скрытых каналов в операционных системах посвящено системам, реализующим мандатную модель доступа. Возможно, это связано с тем, что мандатная модель ставит казалось бы непреодолимые барьеры на пути распространения конфиденциальной информации сверху вниз, которые тем не менее преодолеваются секретными каналами. Рассмотрим несколько примеров скрытых каналов в ОС с мандатным доступом, где процесс/пользователь более высокого уровня допуска (High) пытается нелегально передать конфиденциальную информацию процессу/пользователю более низкого уровня допуска (Low), обходя систему контроля доступа.

Чтение имен файлов или каталогов. Если в системе пользователю Low не разрешено читать содержимое файлов с более высоким уровнем допуска, но разрешено читать

[1] http://www.infosecurity.ru/_gazeta/content/060922/article01.shtml

содержимое каталога, содержащего такие файлы, то скрытый канал может быть организован очень простым способом. Для этого пользователь High может, например:

- давать файлу имя, несущее информацию;
- просто помещать в каталог файл с определённым именем или удалять его, информируя о некотором событии с двумя состояниями (аналог цветка на подоконнике);
- помещать в каталог определенное количество файлов – именно это количество является сообщением;
- придумать еще много способов использовать такую вольность с чтением содержимого каталога для передачи информации на нижний уровень.

Использование факта блокировки файла. Этот вариант скрытого канала был описан в статье Батлера Лэмпсона «Заметка о проблеме ограничения» (A Note on confinement problem), опубликованной в 1973 году в журнале Communications of the ACM[2], в которой впервые было введено понятие скрытого канала, а также были рассмотрены несколько типов таких каналов и пути их предотвращения.

В примере Лэмпсона используется некоторый абстрактный[3] системный вызов открытия файла open(file) со следующими свойствами. Когда какой-либо процесс открывает файл вызовом open(file), то для всех других процессов этот файл становится недоступным – заблокированным. Все попытки других процессов открыть данный файл являются неудачными. Файл становится доступным только после того, когда использующий его процесс выполнит системный вызов закрытия файла - close(file).

. Этим системным вызовом могут пользоваться как High, так и Low процессы. Лэмпсон показал, что с помощью трех несложных процедур, использующих вызовы `open(file)` и `close(file)`, а также трех файлов с низким уровнем секретности, которые процедура более высокого уровня допуска High может открывать только для чтения, а процедура с более низким допуском Low для чтения и для записи, можно организовать передачу данных от High к Low, в обход запретов мандатной системы доступа ОС.

На рис. 7.4 приведена блок-схема процедуры `settrue(file)`, которая циклически пытается открыть файл `file`. В случае успеха – файл разблокирован и открыт - процедура завершает работу.

[2] http://www.cs.umd.edu/~jkatz/TEACHING/comp_sec_F04/downloads/confinement.pdf

[3] то есть, не относящийся к определенной ОС и языку программирования

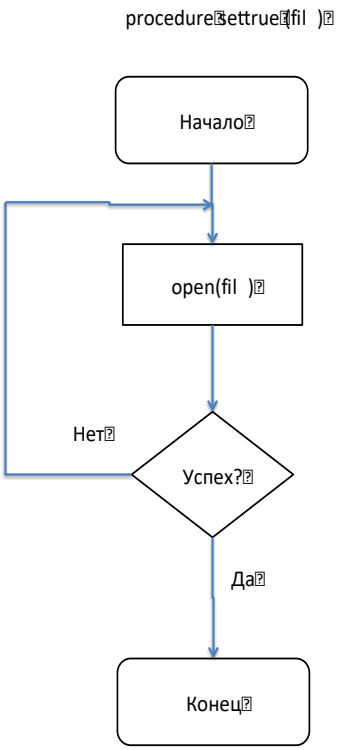

Рис. 7.4. Процедура settrue(file)

Процедура setfalse(file) (рис. 7.5) выполняет закрытие файла file. Предполагается, что она всегда успешна.

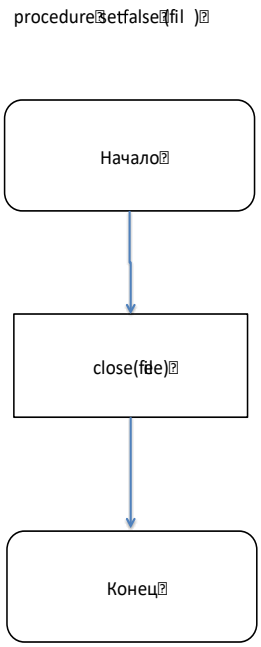

Рис. 7.5. Процедура setfalse(file)

Третья процедура (рис. 7.6) устанавливает значение бита в значение `true` (1), если файл заблокирован (попытка открыть файл оказалась неуспешной), и значение `false` (0) - в противном случае.

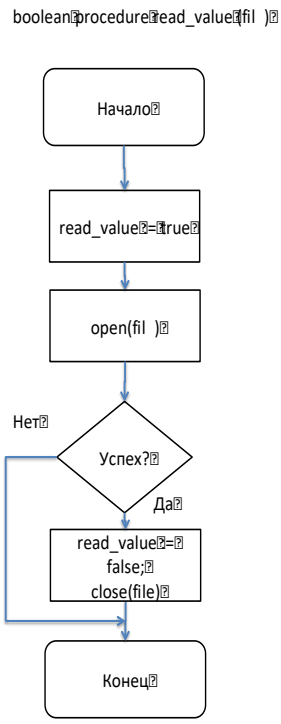

Рис. 7.6. Процедура чтения состояния файла `read_value(file)`

Нетрудно показать, как можно передавать данные с верхнего уровня на нижний, используя три описанные процедуры и три файла: `read_value` для передачи данных, `send_clock` и `receive_clock` для синхронизации процессов передачи и чтения данных. Этот процесс иллюстрируется рис. 7.7.

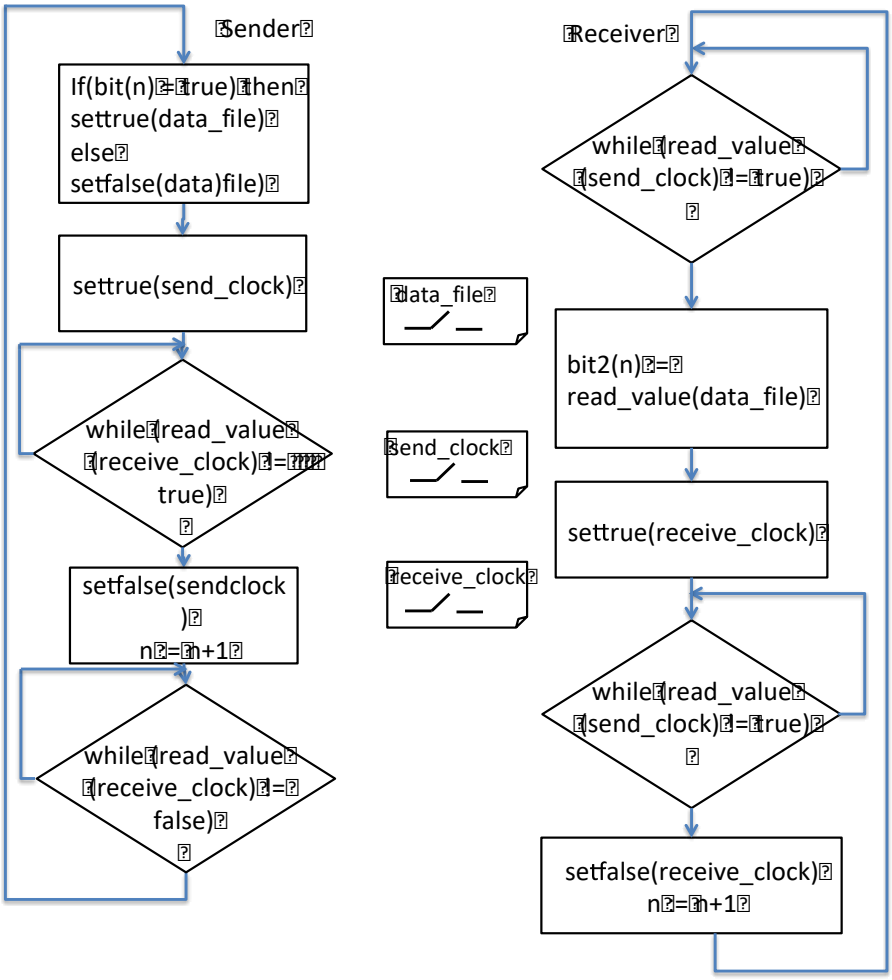

Рис. 7.7. Передача данных между процессом `Sender` (высокий уровень допуска) и `Receiver` (низкий уровень допуска).

Процесс `Sender` читает данные из массива `bit(n)`, которые он хочет передать процессу `Receiver`, используя технику скрытого канала, и открывает файл `data_value`, если бит равен 1 (`true`), или закрывает файл `data_value`, если бит равен 0 (`false`). Затем `Sender` сообщает процессу `Receiver` о том, что очередной бит был передан, устанавливая *сигнализирующий файл* `send_clock` в открытое состояние (`true`).

Процесс `Receiver` ждет, пока файл `send_clock` не перейдет в открытое состояние, после чего считывает состояние файла `read_value` и присваивает соответствующее значение своему массиву `bit2(n)`. После этого он устанавливает *свой сигнализирующий файл* `receiver_clock` в открытое состояние (`true`), оповещая процесс `Sender` о том, что очередное значение массива `bit(n)` прочитано.

Процесс `Sender`, дождавшись сигнала о том, что очередной бит считан, устанавливает файл `send_clock` в начальное состояние `false`, ждет, пока процесс `Receiver` установит свой сигнализирующий файл `receive_clock` в начальное состояние `false` и затем передает аналогичным образом следующий бит массива `bit(n)`.

Скрытый временной канал может быть построен на основе синхронизации обращений к некоторому системному ресурсу, например к сокету TCP или UDP. Процесс-получатель считывает периоды занятости определенного сокета и получает таким образом информацию. В другом случае процесс может кодировать информацию, посылая пакеты на удаленный хост в определенные моменты времени.

Сетевые средства и протоколы предоставляют большой простор для создания скрытых каналов. Например, в пакете TCP SYN имеется *параметр* ISN - начальный номер последовательности (Initial Sequence Number), - которому присваивается произвольное значение при запросе установления TCP-соединения. Вполне логично использовать этот параметр для кодирования скрытой информации, передавая в то же время по установленному соединению легальные сообщения. То же самое можно сказать и о *номерах портов* клиентской части протоколов TCP или UDP, которые также выбираются произвольным образом. Источником информации может также служить *размер пакета*, если дополнять пакеты с легальной информации заполнителем до размера, соответствующего кодовому значению. В последнем случае канал может быть отнесен как к скрытому, так и к стегоканалу, поскольку при передаче скрытой информации обычный информационный поток используется как контейнер.

Ну и нельзя не сказать чуть-чуть подробнее о классическом ***стеганографическом канале***, который использует модификацию битов цифрового изображения. Существует большое количество программ, в том числе и бесплатных (например, EZStego, Xiao Steganography, Steganography Studio), которые умеют делать это для различных форматов цифровых изображений – JPEG, TIFF, BMP, PNG и других. Изменение одного младшего бита цвета RGB-пиксела с 24-битовой палитрой не изменяет восприятие изображения, а информация предается внутри фотографии достаточно большая. Для того чтобы такой канал стал действительно тайным, сама по себе передача фотографий не должна вызывать подозрений, так, отправка фотографий кошек настолько популярна в социальных сетях, что их можно использовать как очень хороший контейнер – только надо делать это периодически, создавая для себя устойчивую легенду члена клуба ми-ми-ми.

Можно встроить скрытую информацию и в текстовые файлы. Например, популярный редактор MS Word использует служебные метки «Начало текста» и «Конец текста», при этом служебная информация, не попадающая в область текста между метками начала и конца, не отображается на экране или печати. Поэтому байты скрытого сообщения можно встроить в служебную информацию незаметным для пользователя способом. Вообще, развитые текстовые редакторы используют большое количество служебных меток, таких как начало нового стиля, метка индекса и т.п., которыми можно кодировать скрытое сообщение.

Борьба со скрытыми каналами

Самым эффективным способом борьбы со скрытым каналом является его уничтожение. Эту идею проще всего проиллюстрировать на примере стегоканала, использующего цифровые изображения. При конвертировании формата скрытые биты скорее всего будут потеряны и сообщение автоматически уничтожено. Поэтому любое конвертирование цифровых изображений является эффективным способом борьбы с этом видом скрытых каналов. То же самое относится и к архивированию, оно применимо и к текстовым файлам.

Уничтожить скрытый временной канал можно сглаживанием трафика за счет внесения задержек пакетов. Такая техника, известная как шейпинг (shaping), давно используется для уменьшения очередей пакетов в буферах маршрутизаторов и коммутаторов.

Скрытый канал, использующий номера портов TCP/UDP, можно разрушить, целенаправленно изменяя эти номера.

Существует теоретический подход, позволяющий выявлять потенциальные скрытые каналы на основе анализа моделей. В обзоре «Скрытые каналы»[4] упоминается две таких модели: модель зависимостей и модель матрицы ресурсов.

Некоторые специалисты по безопасности высказываются скептически в отношении практической важности борьбы со скрытыми каналами, считая, что большой интерес к скрытым каналам объясняется только необходимостью сертификации систем и естественным любопытством ученых, а на практике злоумышленники не используют такие экзотические средства[5].

[4] http://www.jetinfo.ru/Sites/new/Uploads/2002_11.DF9C812FFBD9496BAE9694E27F2D9D1D.pdf

[5] В.А. Галатенко «О каналах скрытых, потайных, побочных. И не только», http://www.jetinfo.ru/stati/o-kanalakh-skrytykh-potajnykh-pobochnykh-i-ne-tolko