Семинар Лыдина С. С. на InfoSecurity-2016

Защита информации как ограничение универсальности устройств

Исторически получилось так, что современные компьютеры являются реализацией идеи «машины Тьюринга», то есть универсального исполнителя. Это означает, что принципиально они созданы так, чтобы выполнить любую задачу. Любую, а не только те, что мы бы хотели.

Мы, пожалуй, довольно далеки от опасений насчет бунта компьютеров и захвата ими власти над миром, но полностью разделяем убежденность в том, что универсальность современных компьютеров является основным источником всех связанных с компьютерной сферой угроз. Если компьютер способен выполнить любую задачу, то он выполнит и вредоносную.

Именно в этом состоит цель мероприятий по обеспечению информационной безопасности, если посмотреть на нее с определенной дистанции: необходимо добиться того, чтобы все наши (то есть легальные) задачи решались, а задачи злоумышленников (нелегальные) – не решались.

Отсюда распространенное убеждение о том, что все задачи защиты информации сводятся к управлению доступом субъектов к объектам. Это не вполне так, к этому сводятся не все задачи.

Если пытаться уложить деятельность по защите информации в некую максиму, то скорее она будет звучать так: защита информации – это ограничение универсальности средств вычислительной техники.

Пытаясь защититься от вредоносных хакерских программ, человечество уже более 60-ти лет разрабатывает программы, традиционно относимые к области защиты информации – средства идентификации, аутентификации, авторизации, контроля целостности, антивирусные программы, криптографические средства и так далее. Использование этих средств отчасти приносит положительный эффект, но только отчасти. Действуя в рамках пусть универсальной, но одной формальной модели, мы неизбежно натолкнемся на ее неполноту.

Становится очевидным, что искать уязвимости только в программном обеспечении явно недостаточно.

Универсальность компьютера обеспечивается архитектурно, самой «конструкцией» машины Тьюринга, как мыслимой в абстракции, так и реализованной на практике.

Поскольку архитектуру нельзя изменить программным путем, то никакие программные средства не помогут нам защититься от хакеров надежно. Игра «кто кого» продолжается уже много лет, давая работу сотням тысяч специалистов по информационной безопасности, но не спасая нас от потерь.

Как же быть?

Если уязвимость в архитектуре – то и совершенствовать нужно архитектуру. И здесь мы подходим к первому, самому радикальному разделению направлений. Мы можем: усовершенствовать архитектуру уже существующих технических средств или использовать новые технические средства на базе новой, более совершенной архитектуры.

Следуя первому направлению, эксплуатирующие организации, приобретая новую технику, устанавливает на нее те или иные средства защиты, следуя второму – приобретают технику, спроектированную тем или иным особым образом.

Наложенные средства защиты информации

Для того чтобы действовать в парадигме утверждения – «компьютер – это только инструмент», необходимо иметь возможность убедиться в том, что это именно Ваш инструмент, а не кого-то, кто его модифицировал для выполнения собственных задач, возможно, далеко не совпадающих с задачами легального пользователя.

Чтобы убедиться в неизменности аппаратной и программной среды компьютера, необходимо провести контрольные процедуры. Однако очевидно, что если контрольные процедуры производятся измененным в свою очередь компонентом, то в них нет никакого смысла.

Именно поэтому контролировать неизменность среды нельзя программными средствами – так как программа может быть изменена. Для того чтобы убедиться, что она не была изменена – ее нужно сначала проверить. Если ее мы проверяем другой программой, то сначала нужно проверить ту программу, которой мы проверяем первую… и так далее. Мы попадаем в зону действия известного парадокса «кто будет сторожить сторожей?». В защите информации попытки контролировать целостность среды программными средствами носит название «синдром Мюнгхаузена», поскольку они аналогичны попыткам вытащить себя самого из болота за волосы.

Продолжая эту аналогию легко прийти к правильным выводам: вытащить себя из болота за волосы – нельзя, потому что нет точки опоры. А вот если тянуть за ветку дерева, растущего на кочке – то можно, потому что у дерева есть точка опоры.

Что может означать «точка опоры» применительно к компьютерной системе фон-неймановского типа (а абсолютное большинство современных настольных компьютеров имеют именно такую архитектуру), не различающей команды и данные, системе, в которой одним из основных действий является «запись», то есть системе принципиально модифицируемой?

«Точка опоры» может означать только одно: контролирующие процедуры должны быть вынесены из этой модифицируемой среды в среду немодифицируемую и легко проверяемую, то есть простую, небольшую по объему (тогда легко обеспечить ее верифицируемость). Это означает аппаратное устройство, независимое от компьютера, который оно проверяет.

Независимость контролирующего устройства – обязательное требование: если часть процедур или решений об обработке их результатов вынесены в основной (контролируемый) компьютер, то модифицированной системой могут быть навязаны любые результаты контроля. Эффект от применения аппаратуры сведется к нулю.

И наконец, самое главное – независимое аппаратное контролирующее устройство должно стартовать первым, до старта операционной системы, иначе у модифицированной системы будет возможность отключить контроллера. «Кто первый встал, того и тапки». Стартовать первым должно то, чему мы доверяем.

Такое аппаратное, простое, независимое от компьютера контролирующее устройство, стартующее первым, до загрузки ОС компьютера – называется резидентный компонент безопасности, РКБ.

Резидентный компонент безопасности – это встроенный в вычислительную систему объект, способный контролировать целостность среды путем сравнения ее параметров с эталонными.

Задача РКБ – сделать так, чтобы на этапе прохождения контрольных процедур защищаемый компьютер не был универсальным, или «машиной Тьюринга», а потом, после их успешного завершения, пользователю снова становились доступны все плюсы универсальности.

Ключевые характеристики РКБ:

- это устройство памяти с очень высоким уровнем защищенности (его внутреннее программное обеспечение должно быть немодифицируемым)

- примитивное (иначе обеспечение его собственной защищенности эквивалентно задаче защиты компьютера, который он защищает)

- встроенное в контролируемую систему и стартующее до старта основной ОС (иначе его функционирование будет необязательным)

- независимое от контролируемой системы (функционирующее автономно)

- перестраиваемое (то есть предполагающее функционирование в режиме управления, когда возможно изменение политик (только специальным привилегированным пользователем) и в пользовательском режиме, когда изменение политик невозможно, и осуществляется только контроль их выполнения).

Концепция РКБ реализована во многих решениях. Каждое из них включает в себя аппаратный компонент (базис) и может включать в себя программную надстройку, неразрывно связанную с этим базисом.

Устройства с правильной архитектурой

Программы и данные, составляющие принадлежащие вам информационные ресурсы, легче сохранить, если компьютер будет защищен. Защита компьютера не гарантия от нарушения целостности программ, но это абсолютно необходимый рубеж защиты.

Казалось бы, в защищенном компьютере ничто не может повлиять на состояние данных и программ – однако же нет! В процессе исполнения одна программа вполне может изменить состояние данных другой программы и даже код другой программы – так, собственно, и ведут себя вирусы, да и не только они.

Именно поэтому на «рабочих» компьютерах, а особенно – на машинах, участвующих в технологических процессах, как правило обеспечена изолированная программная среда, а то и функционально-замкнутая среда. Очевидно, что избежать влияния потенциально опасных программ на функционирование критически важных, тех, которым следует выполняться исключительно корректно, невозможно, не изолируя их одну от другой. Однако, как изолировать клиент-банк на компьютере клиента от его он-лайн игры?

Потери при ДБО в системах класса «Клиент-Банк», «Банк-онлайн» и других всегда связаны с хакерскими атаками – как (возможно и такое) клиентов, так и не-клиентов банка. Банк обычно неплохо защищен, и слабым звеном системы является компьютер клиента. Если компьютер клиента незащищен, то нельзя ни надежно идентифицировать клиента, ни доверять его подписи, так как она устанавливается в этом случае в недоверенной среде.

Надежная (достаточная) защита компьютера стоит около 15,0 – 20,0 тыс. рублей, что неприемлемо для подавляющего числа клиентов. При этом защита ощутимо ограничивает возможности применения компьютера для других целей, что также неприемлемо для клиентов.

Кроме того, существующие системы защиты сложны и требуют специальных знаний для настройки, а этого от клиента требовать вообще невозможно. Более того, если знания такие есть (или еще хуже, пользователю ошибочно кажется, что они есть) – то в своем стремлении улучшить настройки, клиент может завести ситуацию очень далеко от предписанного документацией состояния. Клиента в этом сложно обвинять: со своими устройствами, вообще говоря, он имеет право делать все, что сочтет нужным и сможет.

Таким образом, традиционные подходы к защите информации для систем ДБО непригодны.

Если базовая уязвимость компьютеров – в их архитектуре, значит, компьютер, удовлетворяющий требованию one-touch-security, должен быть создан на основе принципиально иной архитектуры, не имеющей этой уязвимости. Компьютеры, о которых пойдет речь – это компьютеры, архитектура которых отличается и от архитектуры фон-Неймана, и от гарвардской архитектуры.

Отличительной особенностью архитектуры фон-Неймана является то, что команды и данные не разделяются, они передаются по единому общему каналу.

Гарвардская архитектура предполагает наличие разных каналов для команд и данных. Такая схема взаимодействия требует более сложной организации процессора, но обеспечивает более высокое быстродействие, так как потоки команд и данных становятся не последовательными, а параллельными, независимыми.

Однако, и в случае компьютера фон-Неймановского типа, и компьютера с Гарвардской архитектурой организация потоков команд и данных таковы, что архитектурная уязвимость присуща каждому из них. Гибкость, универсальность и в одном, и в другом случае обеспечивается возможностью изменения последовательности команд и данных (двунаправленные стрелки от процессора к памяти) – независимо от того, в одной памяти они лежат, или разделены. В свою очередь, возможность изменения последовательности команд и данных создает и возможность для несанкционированного вмешательства вредоносного программного обеспечения – это и есть основная архитектурная уязвимость, на которой базируются атаки на «перехват управления», описанные в предыдущем разделе.

Однако, если в компьютерах, использующих Гарвардскую архитектуру, где потоки команд и данных уже разделены, сделать память неизменяемой, то не будет необходимости использовать сложные механизмы контроля целостности программ и данных до старта ОС, а контрольные процедуры в этом случае можно исполнять под управлением проверенной и неизменяемой ОС.

Очевидно, что такая архитектура обеспечит неизменность ОС, программ и данных. Такой компьютер приобретет значительный «вирусный иммунитет», так как вредоносное ПО не будет фиксироваться на компьютере.

Недостатком при этом будет то, что придется дорабатывать практически все программное обеспечение, так как разработчики существующего ПО не ограничивают себя в использовании операций записи в память. Для работы практически всех программ необходима возможность записи.

Для того, что бы можно было использовать без доработок все ранее разработанное ПО, необходимо предложенную архитектуру дополнить блоками сеансовой памяти – в которой и будут исполняться программы.

Таким образом, архитектура компьютера будет отличаться на разных этапах – это и есть динамически изменяемая новая гарвардская архитектура.

Она отличается тем, что в ней используется память, для которой установлен режим «только чтение». При загрузке команды и данные размещаются в сеансовой памяти, в которой и исполняются. Начальная загрузка и копирование кодов в сеансовую память могут выполняться как последовательно, так и параллельно – суть разделения этапов от этого не меняется.

Новая архитектура характеризуется динамической изменяемостью, что обеспечивает защищенность и эффективность, неизменность операционной системы, «вирусный иммунитет», и не мешает возможности применения адаптированных стандартных ОС и всего программного обеспечения, написанного для них.

Основных преимуществ два – высокий уровень «вирусного иммунитета» и возможность создания и поддержки доверенной среды, возможность использовать все ранее наработанное программное обеспечение.

Важно то, что на основе описанной архитектуры можно создавать компьютеры для всех видов информационного взаимодействия, при которых доверенность и защищенность взаимодействия важна – от ДБО и защищенных «облаков» до «интернета вещей».