Десницкий В. А., Котенко И. В.

ПОДХОД К ЗАЩИТЕ ПРОГРАММ НА ОСНОВЕ МЕХАНИЗМА УДАЛЕННОГО ДОВЕРИЯ

Санкт-Петербург, СПИИРАН
desnitsky@comsec.spb.ru, ivkote@comsec.spb.ru

Работа посвящена разработке и анализу модели защиты программ на основе механизма удаленного доверия. Цель данного подхода – обнаружение несанкционированных модификаций клиентской программы, выполняющейся в потенциально враждебном окружении. Механизм ориентирован, в первую очередь, на защиту приложений, для корректного функционирования которых требуются сетевые коммуникации с удаленными клиентами или серверами. Тем самым, в случае обнаружения неразрешенных модификаций, появляется возможность уведомления всех сетевых сущностей, с которыми данный клиент осуществляет коммуникацию, что его программное обеспечение взломано. В результате, в случае обнаружения взлома появляется возможность прекратить предоставление программе сетевых сервисов, а также всякую поддержку, необходимую для ее корректной работы.

В соответствии с настоящим подходом [1-5], предполагается, что имеются, требующая защиты, клиентская программа, выполняющаяся в рамках ненадежного окружения, и доверенный сервер, расположенный на защищенном хосте, которые способны осуществлять постоянную сетевую коммуникацию. Одним из важнейших принципов механизма удаленного доверия является реализация, загружаемого во время выполнения и периодически обновляемого, программного модуля. Основное назначение модуля – выполнение верификаций клиентской программы с целью обнаружения злонамеренных модификаций. Регулярное замещение модуля его новой версией позволяет повысить устойчивость программы к взломам, ограничив время, доступное злоумышленнику. Более конкретно, механизм замещения предотвращает возможность неограниченного по времени использования взломанной программы, вынуждая злоумышленника после каждого замещения модуля заново применять необходимые методики взлома. Каждая новая версия модуля отличается обновленными функциями верификации, а также новым набором криптографических ключей. Основное требование к реализации механизма замещения состоит в том, чтобы время, затрачиваемое нарушителем на взлом модуля, не уменьшалось бы ниже определенного предела для всех последующих версий модуля, при условии, что нарушитель смог взломать предыдущие версии модуля.

В работе представлены основные виды атак, компрометирующие защищаемую программу, в том числе, атака клонирования, которая является наиболее сложно обнаружимой. Рассматриваются дополнительные идеи, которые могут использоваться для повышения устойчивости программы к атакам, а именно, осуществление части верификаций в пределах доверенного сервера; реализация модели с несколькими мониторами, каждый из которых верифицирует остальные; само-верификация монитора. Для усложнения атак на программу, и тем самым для увеличения времени реализации атак, программа также может защищаться любыми другими методами и приемами защиты, доступными в настоящее время. В частности, на стороне клиента возможно использование дополнительного аппаратного обеспечения, например смарт-карт, позволяющего выполнять наиболее важные с точки зрения безопасности вычисления более безопасным образом.

Для построения механизма замещения предложено использовать технологии, реализующие парадигму динамического аспектно-ориентированного программирования [5].

Одним из важнейших условий, необходимых для успешной реализации механизма защиты на практике, является его масштабируемость. В целом же наиболее критически важными являются ресурсы доверенного сервера, которые используются для обслуживания большого числа клиентов. В настоящее время в работе рассматривается два серверных ресурса: затрачиваемое процессорное время и объем необходимой памяти.

Задача масштабируемости по сути сводится к поиску некоторого условного компромисса между суммарной степенью защищенности и производительностью системы. Причем, в данной ситуации ошибочным будет представление, что некоторая часть безопасности приносится в жертву ради достижения целей производительности. Скорее, исходя из практической применимости, конкретный программный или программно-аппаратный метод не будет считаться приемлемым, если он не удовлетворяет требованиям масштабируемости.

В рамках рассматриваемого подхода по защите программ предполагается использование различных программных и программно-аппаратных методов. В частности, предлагается использовать различные методы обфускации, криптографические схемы на основе White-Box Cryptography и Physically Observable cryptography, методы Dynamic Memory Placement, Crypto guard, Orthogonal replacement и другие. Каждый такой метод, будучи реализованным в качестве элемента механизма, предоставляет определенный вид защиты программы.

Реализация большинства таких методов разделена между клиентской и серверной сторонами. Так, например, на клиенте происходит вычисление некоторых контрольных сумм, корректность которых затем проверяется на доверенном сервере. Поэтому, вследствие требования масштабируемости механизма, возникает задача минимизации подобных серверных вычислений. В противном же случае, когда для каждого клиента доверенный сервер должен регулярно осуществлять вычисления, требующие значительных затрат, поддержка большого числа клиентов может оказаться проблематичной и практически неосуществимой.

В работе проводится оценивание используемых в механизме защиты методов на предмет их затратности по отношению к потребляемым на доверенном сервере ресурсам. Для этих целей для каждого из методов определяются метрики, позволяющие оценить величину процессорного времени сервера, а также размер памяти, необходимой для его выполнения. Такие метрики могут быть определены теоретически, например, на основе анализа наиболее вычислительно сложных алгоритмов, вовлеченных в реализацию данного метода. В результате может быть получена зависимость загрузки центрального процессора на единицу времени как от размера и свойств клиентской программы, так и от параметров самого метода. Метрика может быть построена также эмпирически – на основе экспериментальных исследований, в частности измерения величины ресурсов при реализации программного прототипа метода защиты.

После того, как для каждого метода выявлены характеристики, связанные с его требованиями к размеру ресурсов сервера, для каждого метода определяется степень предоставляемой им защиты. Иначе говоря, поскольку суммарный ресурс сервера ограничен, то разумно выбрать для реализации лишь “наиболее важные” из методов.

Так, предполагается определять степень защиты, оценивая сложность выполнения эффективных атак на каждый из рассматриваемых методов. Однако данная задача представляется очень сложной или даже практически не осуществимой, так как требует создания формального представления для каждого из методов и атак на него. Задача осложняется еще и тем обстоятельством, что каждый из методов может иметь большое количество параметров, что значительно усложняет анализ.

Поэтому может использоваться более простой подход: выстроить все предложенные методы в последовательность по уменьшению степени предоставляемой ими защиты. В результате чего при ограниченном ресурсе сервера определение конкретных методов,

которые следует выбрать, сводится к задаче динамического программирования (“задача о рюкзаке”).

Актуальной является и обратная задача. Исходя из заданного набора методов, определить максимально допустимое количество клиентов, которое конкретный доверенный сервер способен обслуживать корректным образом.

Предлагается также идея разработки методики подразделения клиентов на группы, для каждой из которых выбирается свой набор реализуемых на клиенте и поддерживаемых сервером методов, имеющий свою ресурсоемкость и степень защиты. В такой ситуации вводится понятие репутации клиентов, в соответствии с которым, в случае если клиент длительное время работал корректно, то можно (возможно, на некоторое время) ослабить степень защиты его копии программы от вмешательств с его стороны. Можно также делить клиентов на группы и по каким-то другим признакам, например, исходно, при их регистрации на основе каких-либо внешних данных.

Ослабление защиты подразумевает в конечном итоге, что клиент сможет осуществить атаку и воспользоваться ее результатами до наступления следующей итерации процедуры замещения. Тем не менее, такую, пусть и ослабленную защиту, все равно имеет смысл применять, так как в случае совершения атаки, она все равно будет обнаружена (хоть и несколько позже), и данный клиент перестанет считаться аутентифицированным (то есть корректным).

Дополнительно, предполагается построение методики динамического изменения текущего набора методов защиты, без приостановки и перезапуска системы. Каждое такое изменение может реализовываться в рамках последующих итерации процедуры замещения.

В настоящее время также ведется работа по созданию программного прототипа механизма, который позволил бы администратору или проектировщику системы найти индивидуальный компромисс между уровнем защищенности и производительностью системы.

Работа выполнена при финансовой поддержке РФФИ (проект №07-01-00547), программы фундаментальных исследований ОНИТ РАН и при частичной финансовой поддержке, осуществляемой в рамках проекта Евросоюза RE-TRUST (контракт № 021186-2).

Литература

1. RE-TRUST project. http://www.re-trust.org
2. Ceccato M., Ofek Y., Tonella P. Remote entrusting by run-time software authentication // Proceedings of the International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM 2008). Tatras, Slovakia. 2008.
3. Десницкий В.А., Котенко И.В. Защита программного обеспечения на основе механизма “удаленного доверия” // Изв. вузов. Приборостроение, Т.51, № 11, 2008.
4. Десницкий В.А., Котенко И.В., Резник С.А. Разработка и верификация протокола обмена сообщениями для защиты программ на основе механизма “удаленного доверия” // Защита информации. Инсайд, № 4, 2008. С.59-63; № 5, 2008. С.68-74.
5. Котенко И.В., Десницкий В.А. Аспектно-ориентированный подход к реализации мобильного модуля в модели защиты, основанной на механизме “удаленного доверия” // Информационные технологии и вычислительные системы, № 5, 2008.