ЭЛЕКТРОНИКА, РАДИОТЕХНИКА И СВЯЗЬ

УДК: 004.056.5

РАЗРАБОТКА НОВЫХ ПОДХОДОВ ДЛЯ ОБЕСПЕЧЕНИЯ БЕЗОПАСНОСТИ ИНФОРМАЦИОННЫХ СИСТЕМ, ИСПОЛЬЗУЮЩИХ ОТКРЫТЫЕ ПРОТОКОЛЫ

Александр Олегович НЕВОЛИН родился в 1983 г. в городе Ялте. Ассистент МАИ. Основные научные интересы — в области информационных технологий. Автор семи научных работ. E-mail: nao@nevlabs.ru.

Alexander O. NEVOLIN, was born in 1983, in Yalta. He is an Assistant Professor at the MAI. His research interests are in information technologies. He has published 7 technical papers. E-mail: nao@nevlabs.ru

В статье описываются открытые протоколы, область их применения, достоинства и недостатки. Рассматриваются существующие способы обеспечения безопасности при их применении, анализируются недостатки этих способов. Предлагаются новые решения для повышения защищенности передаваемых данных, оценивается их теоретическая эффективность. Приводятся результаты эксперимента по оценке их эффективности и производительности.

Open XML-based protocols are described with their application areas as well as their intrinsic virtues and shortcomings. Existing techniques are analyzed to support information security when these protocols are used. Disadvantages are revealed for these techniques. New approach is suggested to enhance security of transferred data basing on the protocols and its theoretical efficiency is evaluated. Test results are presented to demonstrate efficiency of the approach.

Ключевые слова: информационная безопасность, XML, запутывание, обфускация, шифрование, информационное взаимодействие, протоколы.

Key words: information security, XML, obfuscation, encryption, data-driven interaction, protocols.

1. Актуальность проблемы

В настоящее время стремительно развиваются распределенные информационные системы. Наравне с задачей организации выполнения каждым из компьютеров своей собственной работы важной задачей является обеспечение взаимодействия компьютеров между собой. Сетевые технологии продолжают интенсивно развиваться, и наибольший прогресс заметен в верхнем уровне модели OSI — прикладном. В то время как протоколы более низких уровней практически устоялись и меняются незначительно, на прикладном уровне постоянно появляются новые протоколы и принципы взаимодействия. Зачастую это вызывает некоторые проблемы обеспечения совместимости различных приложений: многие компании-разработчики программного обеспечения (ПО) используют свои собственные бинарные протоколы, и для организации взаимодействия с продуктами других компаний приходится тратить значительные усилия на программирование модулей протоколов.

Кроме того, вышеописанная проблема усугубляется тем, что все большее распространение получают так называемые гетерогенные сети — сети, в которых присутствуют компьютеры с различными типами операционных систем (например, одновременно с ОС семейства Windows и Unix). Своевременная поддержка протоколов, работающих в бинарном (двоичном) режиме, в таких условиях становится еще более сложной и дорогостоящей.

В то же время относительно недавно появилось некоторое множество протоколов, основанных на XML. Общее описание стандарта XML дано в [1].

XML-протоколы обладают следующими достоинствами перед бинарными протоколами:

• **открытость.** XML — сам по себе открытый формат хранения данных, т.к. вся информация представляется в текстовом виде (теги, содержимое и т.п.);

• **кроссплатформенность.** В настоящее время XML поддерживается всеми распространенными операционными системами. В любой из них имеется встроенный XML-парсер;

• **высокоуровневость.** Разработчикам программного обеспечения не нужно самим создавать парсер XML-документов, так как он встроен в большинство операционных систем. Это также исключает затраты на тестирование, отладку и т.п., так как такой парсер можно использовать как «черный ящик».

Вместе с тем открытость также вызывает проблемы с безопасностью информационного взаимодействия.

Это связано с тем, что открытые, «самодокументированные», данные легко интерпретировать как легальному пользователю системы, так и злоумышленнику. Например, если в системе используется бинарный протокол, неизвестный атакующему, то ему тяжело найти нужную информацию в двоичном потоке данных (например, номер кредитной карты, так как поток может состоять из большого числа цифр). В то же время он сможет легко найти эту информацию в XML-документе.

Таким образом, открытость протокола является достоинством, с точки зрения легальных пользователей, но недостатком с точки зрения объема передаваемых данных и безопасности.

В настоящее время существуют частичные решения некоторых проблем, которые будут рассмотрены далее.

Шифрующие протоколы

Для защиты передаваемых данных можно использовать различные протоколы, поддерживающие шифрование. Наиболее распространенным протоколом, применяемым для передачи гипертекстовых данных (в том числе и XML), является SSL (Secure Socket Layer). При его использовании создается защищенное соединение между клиентом и сервером.

Технология XML-Security

В отличие от шифрующих протоколов, которые шифруют весь трафик полностью, технология XML Security позволяет защитить данные, передаваемые с помощью XML, сохраняя эти данные в виде XML. В настоящее время XML Security поддерживается на встроенном уровне в инфраструктуре .Net Framework, в веб-сервере Apache и др. [2]

Однако шифрующие протоколы и технология XML Security имеют некоторые недостатки:

• использование шифрования само по себе вызывает значительное увеличение трафика. Это приводит к появлению «накладных» расходов на эксплуатирование информационной системы, а в системах реального времени (например, обмена сообщениями) увеличиваются задержки в связи с необходимостью передачи большего объема информации, нежели по незащищенным каналам;

• шифрование представляет из себя набор сложных математических преобразований, что требует значительных вычислительных ресурсов. Особенно данная проблема актуальна в системах, где данные передаются по множеству параллельных логических каналов. В таких системах к аппаратному обеспечению предъявляются высокие требования, что вызывает увеличение стоимости этих систем в целом.

2. Альтернативный подход к обеспечению безопасности данных, передаваемых с помощью XML-протоколов

В статье предлагается альтернативный подход, позволяющий повысить защищенность данных, передаваемых с помощью XML-протоколов.

Суть подхода заключается в выполнении с XML-документом преобразований, аналогичных тем, которые применяются для запутывания промежуточного кода (обфускации) в Java или .NET. Для этого используются переименования методов, классов, переменных, а также запутывания графа потока управления. Кроме того, возможно использование дополнительных приемов, например перегрузки разных методов. Это позволяет еще сильнее запутать программу и усложнить ее анализ злоумышленнику.

Аналогичный подход можно применить и для преобразования XML-документов, которое далее будем называть *запутыванием XML-документов.*

Можно предложить следующие способы запутывания:

1. Переименование названий тегов и атрибутов.
2. Изменение положения листьев и ветвей в XML-дереве.
3. Вертикальная перестановка тегов в XML-дереве.
4. Горизонтальная перестановка тегов в XML-дереве.

5. Композиция/декомпозиция тегов в XML-дереве.

Более подробно эти способы описаны в [3].

3. Критерии эффективности запутывания XML-документов

Пусть имеется информационная система, в которой присутствуют две стороны, обменивающиеся данными. Передача данных осуществляется через открытый канал связи без ошибок. В системе используется протокол, основанный на XML.

Пусть имеется N протоколов, основанных на XML: $Prot_1$, $Prot_2$, ..., $Prot_N$.

Каждый из них состоит из тегов с названием T_1, T_2, ..., T_r, которые образуют алфавит названий тегов A_t, состоящий из r элементов, и атрибутов с названием $Attr_1$, $Attr_2$, ..., $Attr_s$, которые образуют алфавит названий атрибутов A_{attr}, состоящий из s элементов.

Тег с одним и тем же названием в протоколе может использоваться более чем один раз, равно как и атрибут. Таким образом, каждый протокол обладает некоторыми статистическими свойствами:

- вероятность появления каждого тега P_{T1}, P_{T2}, ..., P_{Tr} ;
- вероятность появления каждого атрибута P_{ATTR1}, P_{ATTR2}, ..., P_{ATTRs}.

Запутывающим преобразованием будем называть такое обратимое преобразование Q, которое преобразует исходный XML-документ X в документ X` (X -> X`).

Будучи примененным к какому-либо протоколу, такое преобразование изменяет его статистические характеристики:

- у существующих тегов могут измениться вероятности появления (вплоть до нуля — это означает, что в запутанном документе такой тег не встречается);
- могут появиться новые теги. Их можно интерпретировать как теги, которые изначально присутствовали в алфавите, но до запутывания вероятность их появления была равна нулю.

Атакующий знает, что сообщения могут быть запутаны, однако точно этот факт ему неизвестен. Цель злоумышленника — извлечение нужной ему информации из передаваемых XML-документов. Если он обнаруживает факт запутывания, то сначала выполняет декодирование, а затем интерпретирует документ. Если факт запутывания не установлен, атакующий сразу приступает к интерпретации документа. Таким образом, для информационной системы оптимальный вариант — необнаружение факта запутывания при его реальном применении.

Как было сказано ранее, защищающаяся сторона применяет алгоритм запутывания, который, в свою очередь, изменяет статистические свойства протокола. Также было сказано, что алгоритм запутывания меняется с определенным интервалом. Следовательно, с тем же интервалом меняются и статистические свойства перехватываемых атакующим сообщений. Из этого вытекает, что если злоумышленник будет иметь две выборки — одну до момента смены алгоритма запутывания, вторую — после — и установит, что они не принадлежат одному и тому же закону распределения с одинаковыми параметрами, то для него будет очевиден факт смены алгоритма запутывания, а следовательно, и факт его применения как такового.

Для выявления факта соответствия/несоответствия двух серий перехваченных данных одному и тому же закону распределения с одинаковыми параметрами злоумышленник может воспользоваться аппаратом из математической статистики — критерием согласия. Наиболее часто применяется критерий Пирсона (Хи-квадрат). Этот критерий можно использовать для проверки не только гипотезы о принадлежности результатов опыта какому-либо известному распределению, но и гипотезы о принадлежности результатов одной выборки результатам другой выборки.

Допустим, перехвачено s серий (выборок) сообщений системы (в нашем случае $s = 2$), каждая серия состоит из n_j, $j = (1, r)$, тегов. На основе перехваченных сообщений злоумышленник находит частоты появления каждого из тегов $p_j^* = n_j / n$ (n_j — число появлений тега; n — общее число тегов в серии) в каждой из серий.

Требуется проверить гипотезу H_0 о том, что во всех сериях (выборках) наблюдалась одна и та же совокупность вероятностей p_j^*.

1. Для каждой i-й серии, $i = (1, s)$, подсчитываются числа n_{ji} появлений тегов T_j, $j = (1, r)$.

2. Подсчитывается суммарное число N_j появлений тега T_j, $j = (1, r)$, во всех сериях, а также числа n_i, $i = (1, s)$ и n:

$$N_j = \sum_{i=1}^{s} n_{ji};$$

$$n_i = \sum_{j=1}^{r} n_{ji};$$

$$n = \sum_{i=1}^{s} \sum_{j=1}^{r} n_{ji}.$$

3. Вычисляется значение z-статистики критерия хи-квадрат по формуле

$$z = \varphi(z_n) = n \cdot \left(\sum_{i=1}^{s} \sum_{j=1}^{r} \frac{{n_{ji}}^2}{n_i \cdot N_j} - 1 \right).$$

4. Как было сказано выше, при больших n распределение статистики z хорошо аппроксимируется с распределением хи-квадрат $X^2(m)$ с $m = (s-1)(r-1)$ степенями свободы.

Формируется критическая область

$$\overline{G} = \left(x_{1-\alpha}(m), +\infty \right),$$

где $x_{1-\alpha}(m)$ — квантиль уровня 1-а распределения $X^2(m)$; α — вероятность ошибки первого рода.

5. Принимается решение: отклонить гипотезу H_0, если $\varphi(z_n) \in \overline{G}$, и принять гипотезу H_0, если $\varphi(z_n) \in G$ [4].

При принятии решения существует некоторая вероятность допущения одной из двух ошибок: ошибка первого рода («ложная тревога» — отклонение гипотезы при том, что она верна) и второго рода («пропуск цели» — принятие гипотезы при том, что она ложна). Фактически решение о принятии или отклонении гипотезы принимается на основе сравнения полученного значения статистики z с некоторым порогом. В соответствии с п. 4 порог выбирается исходя из заданной вероятности ошибки первого рода.

Из математической статистики известно [4], что с понижением порога (сужением доверительной области) повышается вероятность ошибки первого рода (фактически критерий становится более «строгим»), а с повышением порога (расширением доверительной области) повышается вероятность ошибки второго рода (критерий может получиться слишком «мягким»).

В таком случае злоумышленник может применить критерий Неймана—Пирсона, используемый в радиолокации: выбрать такое оптимальное решение, которое обеспечивает минимум вероятности пропуска цели (P_2) при условии, что вероятность ложной тревоги не больше некоторого заданного значения. Будучи примененным к текущей задаче, этот критерий предполагает следующее: злоумышленник выбирает максимально допустимую для него вероятность ложной тревоги и устанавливает порог в соответствии с этой вероятностью. Понижение порога вызовет превышение приемлемой вероятности ложной тревоги, а повышение порога будет неоптимальным, так как повысится вероятность пропуска цели (при том, что вероятность P_1 будет по-прежнему устраивать атакующего).

Таким образом, задача защищающейся стороны — выбрать такие алгоритмы запутывания, которые обеспечили бы максимальную вероятность ошибки злоумышленника, т.е. изменяющие XML-документ таким образом, чтобы отличия его статистических свойств от статистических свойств исходного документа были минимальны.

Работу оптимальной системы запутывания можно представить следующим образом.

Имеется некоторый набор запутывающих преобразований Q_1, Q_2, ..., Q_k, причем некоторые из них могут применяться с различными параметрами. Каждый из алгоритмов (или их комбинация) обеспечивает некоторое значение z-статистики Хи-квадрат, характеризующее степень статистических отличий от исходного XML-документа.

Перед сменой алгоритма запутывания защищающаяся сторона должна выбрать такую комбинацию запутывающих преобразований и их параметров, чтобы обеспечить минимально допустимую эффективность запутывания в идеальном для злоумышленника случае. Подходящий набор алгоритмов может быть выбран на основе двух подходов:

- обеспечение значения статистики Хи-квадрат, меньшего, чем максимально допустимое (заданное в виде порога);
- выбор некоторого фиксированного количества алгоритмов, среди которых выбирается алгоритм, обеспечивающий минимальное значение статистики Хи-квадрат среди остальных. Как вариант, фиксируется не количество алгоритмов, а время, затрачиваемое на поиск оптимального.

В целом выбор конкретного подхода зависит от класса систем, от модели, которую использует злоумышленник и т.п. Таким образом, выбор конкретной модели защищающегося и атакующего предполагается предоставить конечным пользователям в зависимости от реально используемой информационной системы.

4. Апробация предлагаемых методов

Был проведен эксперимент по оценке эффективности различных алгоритмов запутывания между собой на реальных XML-документах.

В качестве примера рассмотрим запутывание XML-документов с помощью переименования тегов. Были использованы следующие параметры:

- процент тегов для переименования — основной параметр. Можно переименовывать не все теги, а лишь некоторую их часть. При этом чем меньше тегов переименовывается, тем меньше разница по

сравнению с исходным документом. Далее этот параметр будем называть глубиной запутывания;

• тип переименования: по словарю (замена имени на другое имя способом, заданным в конкретной реализации, например, замена на синонимы) или замена имен тегов между собой. Последний способ в некоторой степени схож с перестановками тегов;

• использование перегрузки. Ранее упоминалось, что перегрузка — это механизм, который позволяет двум разным тегам дать одинаковое имя. При этом, однако, должен оставаться некоторый различительный признак. В нашем случае их может быть два: количество атрибутов и их имена. В разработанном модуле присутствует возможность выбора типа перегрузки: различать теги только по количеству атрибутов или по количеству атрибутов и их именам.

В качестве примера приведем результаты запутывания вызова метода по протоколу SOAP. На рис. 1 показан график, иллюстрирующий зависимость значения статистики Хи-квадрат от глубины запутывания. Пунктирной линией обозначен порог для доверительной вероятности 0,95.

Порог был выбран исходя из приведенной ранее формулы. Число степеней свободы для данного документа составило 9 (число возможных исходов, т.е. число тегов минус один).

Как видно, для методов переименования, использующих словарь, допустимая глубина запутывания составляет порядка 90%. В случае большей величины значение статистики Хи-квадрат превышает порог, а следовательно, в этом случае факт запутывания будет обнаружен злоумышленником.

Методы переименования, которые переставляют имена тегов между собой, позволяют использовать 100%-ную глубину запутывания с двукратным запасом.

На рис. 2 приведены результаты замеров производительности различных методов запутывания на нескольких реальных XML-документах.

По оси ординат отложено время на одно запутывание (в миллисекундах). В качестве исходных экспериментальных данных были использованы следующие XML-документы:

• XML-документация исходного кода на платформе .Net (GenXMLDoc.xml);

• одно сообщение пользователя, передаваемое по протоколу XMPP (message.xml);

• запрос на проведение чата в Jabber (message_chat.xml);

• вызов метода по протоколу SOAP (request1.xml).

Xml Security проигрывает по производительности во всех случаях. Кроме того, следует учесть, что методы запутывания полностью реализованы на платформе .Net, т.е. в управляемом коде. В то же время какая-то часть реализованной Xml Security, возможно, создана в откомпилированном коде, который сам по себе работает быстрее, нежели про-

Рис. 1. Зависимость значения статистики Хи-квадрат от глубины запутывания

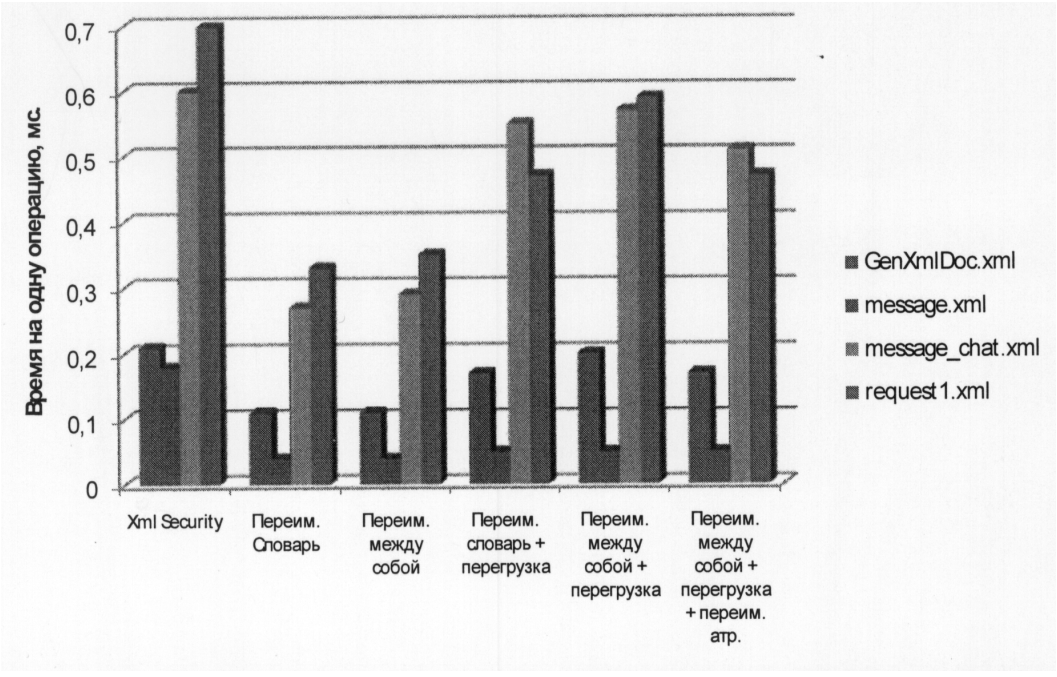

Рис. 2. Результаты замеров производительности различных методов запутывания на нескольких реальных XML-документах

межуточный управляемый код. Таким образом, реальная разница в производительности может быть еще больше, нежели полученная в результате эксперимента.

Выводы

На основании полученных результатов можно рекомендовать использование запутывания XML-документов как альтернативный способ защиты передаваемых данных. Например, в системах, где используется удаленный вызов процедур (веб-сервисы), XML-документы можно эффективно запутывать без риска обнаружения этого факта злоумышленником. В частности, для запутывания SOAP-вызовов можно использовать все методы с глубиной 100%, а простое переименование по словарю — с глубиной порядка 90%. Если учесть, что по словарю можно переименовывать не все теги (некоторые из них входят в стандарт SOAP), то эта цифра вполне приемлема для защищающейся стороны.

В системах обмена сообщения в реальном времени XML-документы также можно эффективно запутывать. Так, например, обычное сообщение от пользователя можно запутывать со 100%-ной глубиной. Более сложные запросы, состоящие из большего числа тегов, подчиняются общей закономерности: переименование между собой в целом более эффективно, нежели переименование по словарю.

Библиографический список

1. *Печерский А.* Язык XML — практическое введение. http://citforum.ru/internet/xml/, дата обращения — январь 2008 г.

2. *Hugo Haas.* XML Security : Signature, Encryption, and Key Management. http://www.w3.org/2004/Talks/0520-hh-xmlsec/, дата обращения — март 2008 г.

3. *Неволин А. О.* Методы повышения безопасности информационного взаимодействия в системах, использующих открытые протоколы // Сб. докладов ежегодной конференции «Информационные технологии и радиоэлектронные системы». 2007. С. 88-93.

4. *Кибзун А.И.* Теория вероятностей и математическая статистика. Базовый курс с примерами и задачами. — М.: Физматлит, 2002.

Московский авиационный институт
Статья поступила в редакцию 24.02.2009