БЕЗОПАСНАЯ ОБРАБОТКА ИНФОРМАЦИОННЫХ ЗАПРОСОВ В РАСПРЕДЕЛЕННЫХ СИСТЕМАХ УПРАВЛЕНИЯ НА ОСНОВЕ СИНТАКСИСА КРИПТОГРАФИЧЕСКИХ СООБЩЕНИЙ (CMS)

Асратян Р.Э.

Институт проблем управления им. В.А. Трапезникова РАН,
Россия, г. Москва, ул. Профсоюзная д.65
rea@ipu.ru

Рассмотрены методы реализации сетевой Службы защищенных сообщений, предназначенной для безопасной обработки информационных запросов в распределенных информационных системах. Отличительными особенностями службы являются тесная интеграция функций информационной защиты данных с функциями информационного взаимодействия в сети. Описаны особенности архитектурного построения службы на основе средств поддержки Синтаксиса защищенных сообщений (CMS) в Windows.

Ключевые слова: распределенные системы, Web-сервисы, Интернет-технологии, информационный обмен, защита данных, информационная безопасность.

Введение

Появление сетевой архитектуры .Net и технологии Web-сервисов [1]) не устранило всех трудностей разработчиков распределенных систем с обеспечением защиты данных в сети. Эти трудности чаще всего бывают связаны с отсутствием в архитектуре .NET встроенных средств защиты и аутентификации сетевых сообщений и более всего проявляются в разработках систем, предназначенных для работы в сложных, мульти-серверных и мульти-сетевых средах в условиях высоких требований к информационной безопасности [2,3].

В работе [4] описана новая сетевая служба PMS (Protected Message Service), разработанная с целью преодоления вышеуказанного недостатка. Суть подхода заключается в тесной интеграции функций сетевого информационного обмена с функциями защиты и аутентификации данных. Внешне эта интеграция проявляется в том, что отмеченные функции входят в набор методов главного программного класса службы – класса «Защищенное сообщение» (PmsMessage), отображающего электронный документ (информационный запрос или ответ), снабженный одной или несколькими удостоверяющими электронными цифровыми подписями (ЭЦП). В отличие, например, от технологии Web-сервисов, описываемая служба опирается не на модель вызова методов удаленных объектов, а на модель обмена сообщениями. В данном случае это означает, что все сервисные обрабатывающие функции (методы) имеют одинаковую, жесткую спецификацию: они получают объект класса «Защищенное сообщение» в качестве параметра и возвращают объект того же класса. Эти обрабатывающие функции группируются в одну или несколько динамических библиотек, которые подключаются к серверу PMS в момент его запуска (каждая библиотека может рассматриваться как отдаленный аналог Web-сервиса в .NET), и становятся доступными для клиентских компонент.

Реализация PMS на основе криптосистемы «КриптоПро» версии 3.6 и проведенные лабораторные эксперименты показали достаточно высокое быстродействие новой службы, не уступающее, а в отдельных случаях превосходящее быстродействие Web-сервисов в одинаковых условиях. Однако, при данном подходе возникает жесткая «привязанность» PMS к определенной криптосистеме, что может создать неудобства для разработчиков распределенных систем.

В данной работе рассматривается новый подход к архитектурному построению PMS, основанный на применении стандарта Cryptographic Message Syntax (CMS) и его программной поддержки в среде Windows в качестве базисного средства реализации. Главное преимущество этого подхода заключается в том, что он позволяет PMS «унаследовать» способность гибкой настройки на использование любой криптосистемы, поддерживающей стандарт CMS, и, тем самым, устранить ту жесткую привязку к определенной криптосистеме, о которой говорилось выше.

1 Методы реализации PMS

На рис. 1 проиллюстрированы два архитектурных подхода к реализации PMS:

- на основе прямого подключения определенной криптосистемы к программным модулям PMS без использования CMS (архитектура «PMS-Криптосистема»),
- на основе CMS (архитектура «PMS-CMS-Криптосистема») с возможностью использования различных криптосистем.

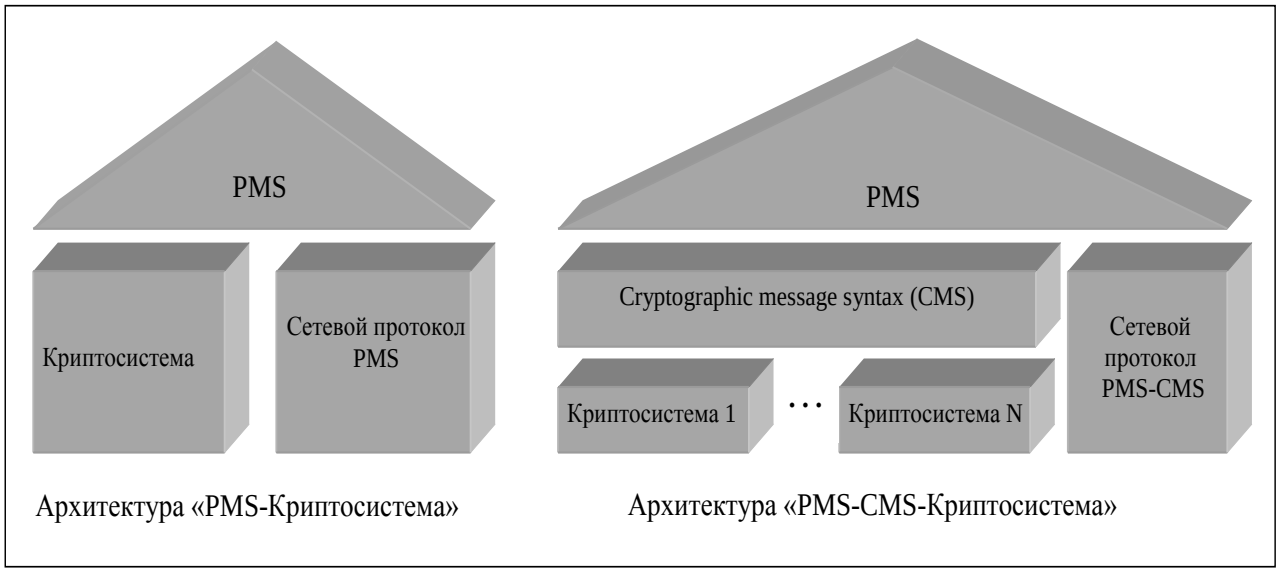

Рис. 1. Архитектурные решения для реализации PMS

Первый подход основан на прямом вызове функций определенной криптосистемы из кода программных модулей PMS для выполнения операций формирования ЭЦП, шифрования и т.п. Очевидно, что этот подход обеспечивает наименьше «накладные расходы», но реализация PMS оказывается жестко привязана к выбранной криптосистеме.

Второй подход основано на использовании средств поддержки CMS в качестве промежуточной прослойки между программными модулями PMS и используемой криптосистемой. Главное преимущество - возможность гибкой настройки на применение любой криптосистемы, поддерживающей стандарт CMS без изменения кода программных модулей службы, которые используют универсальный программный интерфейс CMS для выполнения крипто-функций.

Как видно из рис. 1, важной составной частью каждого из двух архитектурный решений является используемый сетевой протокол. Независимо от выбранного решения, этот протокол организуется по следующим общим принципам.

- PMS в полной мере использует двоичную природу TCP/IP [5]. Взаимодействие между клиентом и сервером PMS осуществляется по специальному, достаточно простому протоколу, ориентированному на передачу двоичных сетевых сообщений (PMS-сообщений) в обоих направлениях (никакие преобразования двоичных данных в текстовую форму типа base64 не применяются). Каждое такое сообщение в общем случае содержит два массива байтов: заголовок сообщения и тело сообщения. Первые 4 байта заголовка или тела сообщения содержат целое число – его длину.
- При передаче запроса от клиента к серверу в заголовок сетевого сообщения помещается строка, содержащая полное имя вызываемой функции, а в тело сообщения упаковывается структура PmsMessage в открытой или зашифрованной форме, содержащая информационный запрос. Строка заголовка используется сервером для организации вызова соответствующей обрабатывающей функции.
- При передаче результата обработки от сервера к клиенту в заголовок сетевого сообщения помещается строка диагностического сообщения (значение параметра Msg, сформированное обрабатывающей функцией), а в тело сообщения упаковывается структура PmsMessage, содержащая ответ сервера в открытой или зашифрованной форме, предварительно подписанный собственным закрытым ключом сервера. Никакие двоично-текстовые преобразования (типа base64) не применяются. Полученное от сервера диагностическое сообщение автоматически присваивается члену ErrMsg объекта класса PmsSrvLibraries на стороне клиента (см. рис. 1).

Тем не менее, в архитектуре «PMS-Криптосистема» детали реализации сетевого протокола PMS могут различаться в зависимости от выбранной криптосистемы, т.к. от последней зависит состав и структура крипто-данных, включенных в подписанный и/или зашифрованный объект PMSMessage в теле сетевого сообщения. Важнейшее преимущество архитектуры «PMS-CMS-Криптосистема» заключается в том, что в этом случае реализация протокола не зависит от применяемых криптосистем

и целиком основывается на универсальном стандарте представления крипто-данных в CMS (RFC 5652).

2 Временные оценки

Для исследования свойств описанного подхода была проведена серия экспериментов с новой версии PMS. Основная цель этих экспериментов заключалась в сравнении быстродействия двух методов ее реализации (на основе архитектур «PMS-Криптосистема» и «PMS-CMS-Криптосистема») между собой, а также с быстродействием Web-сервисов в одинаковых условиях. Главное внимание уделялось вызовам сервисных функций с относительно малым (от нескольких миллисекунд до нескольких сотен миллисекунд) временем выполнения (при более длительной обработке разница между двумя технологиями практически нивелируется) с применением средств ЭЦП и шифрования сообщений на основе криптосистемы «КриптоПро» версии 3.6, соответствующей требованиям действующих в России ГОСТов в области криптографической защиты информации. На рис. 2 показаны характерные результаты экспериментов с очень быстрой сервисной функцией, выполняющей простое перекодирование полученного строчного сообщения в верхний регистр и возврат результата клиенту, при длине сообщения в 2 Кбайт, 50 Кбайт и 100 Кбайт соответственно. На рисунке приведена диаграммы времен выполнения операции на сервере в реализации «PMS-КриптоПро» (черный столбик), в реализации «PMS-CMS-КриптоПро» (серый столбик) и с помощью Web-сервиса (белый столбик). В каждом режиме время выполнения вычислялось, как среднее значение для 100 последовательных вызовов сервисной функции.

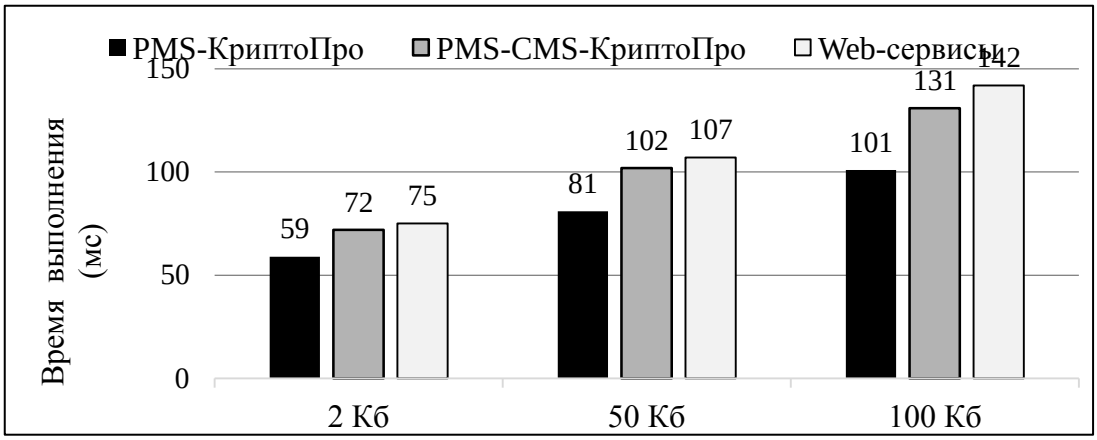

Рис. 2. Оценки быстродействия двух реализаций PMS и Web-сервиса

В целом результаты экспериментов показывают, что реализация PMS на основе архитектуры «PMS-CMS-КриптоПро» в целом несколько уступает в быстродействии «прямой» реализации на основе архитектуры «PMS-КриптоПро. В то же время обе реализации PMS не уступают в быстродействии Web-сервисам.

Литература

1. *Мак-Дональд М., Шпушта М*. Microsoft ASP.NET 3.5 с примерами на C# 2008 и Silverlight 2 для профессионалов. – М.: Вильямс, 2009. – 1408 с..
2. *Згоба А.И., Маркелов Д.В., Смирнов П.И.* Кибербезопасность: угрозы, вызовы, решения / Вопросы кибербезопасности, 2014, № 5. С.30 – 38.
3. *Козлов А.Д., Орлов В.Л.* Методы и средства обеспечения информационной безопасности распределенных корпоративных систем. – М.: ИПУ РАН, 2017. – 156 с.
4. *Асратян Р. Э.* Интернет-служба защищенной обработки информационных запросов в распределенных системах // Программная инженерия, 2016, № 11. С.490 – 497.
5. *Хант К.* TCP/IP. Сетевое администрирование. – СПб.: Питер, 2007. – 816 с.