УДК 681.3

И.Д. ГОРБЕНКО, д-р техн. наук, Е.Г. КАЧКО, канд. техн. наук, С.А. КОВАЛЕВ, асп.

СЕРВИС АУТЕНТИФИКАЦИИ KERBEROS

Важное место при построении систем защиты информации занимает сервис аутентификации Kerberos. В данной работе выполнен обзор возможных протоколов и предлагается программная модель такого сервиса.

Рассмотрим назначение и функции Kerberos [1-7].

Имеется открытая, незащищенная сеть. В ее узлах находятся субъекты: сервера и клиенты. (По определению: сервер предоставляет услуги клиентам; клиент – программное обеспечение, пользователи и др.)

Необходимо подтвердить подлинность (аутентичность) клиента С серверу S и наоборот.

При этом используется следующая предпосылка: у каждого субъекта есть собственный секретный ключ. Для пользователей (под английским термином – User – понимается физическое лицо, в отличие от термина Client, под которым понимается субъект, потребляющий услуги) специфицируется алгоритм преобразования пароля в секретный ключ. Подлинность субъекта отождествляется со знанием его секретного ключа.

Субъект А может подтвердить свою подлинность субъекту В, просто переслав ему свой секретный ключ КА. Но сеть незащищена – ключ могут перехватить, исказить. Требуется менее прямолинейный способ демонстрации знания своего секретного ключа.

Эту задачу выполняет Kerberos – третья сторона, которой доверяют все субъекты, и, соответственно, которая хранит секретные личные ключи всех субъектов. Kerberos не полагается на:

- средства аутентификации, реализованные в операционных системах сетевых компьютеров;
- на подлинность сетевых адресов;
- на физическую защищенность всех сетевых компьютеров.

Kerberos производное от Cerberus. Цербер – трехголовый пес, охраняющий вход в царство мертвых. Сервер аутентификации Kerberos основан на протоколе аутентификации третьей стороны Нидхама (Needham) и Шредера (Schroeder) (1978) с изменениями Дэнинга (Denning) и Сако (Sacco) (1981). Проектирование и реализация с 1-й по 4-ю версии Kerberos'а осуществлялась в рамках проекта Athena Стивом Миллером (Steve Miller) из Digital Equipment Corporation и Клиффордом Ньюменом (Clifford Neuman) (сейчас в Information Sciences Institute of University of Southern California). А также принимали участие и многие другие, в том числе: Джером Салтцер (Jerome Saltzer) - технический директор проекта Athena и Джефри Шилер (Jeffrey Sciller) - менеджер сети университета MIT (Massachusetts Institute of Technology). Над пятой версией работает команда разработчиков MIT во главе с Теодором Тсо (Theodore Ts'o).

Кроме доказательства аутентичности субъектов Цербер позволяет распространить сеансовый и субсеансовый ключи.

Основная идея Kerberos такова: очень сложно из зашифрованного текста извлечь ключ шифрования, также сложно расшифровать закрытое сообщение (без знания секретного ключа). Отсюда идея: послать нужную субъекту информацию, зашифровав ее на его секретном ключе.

1. $K \rightarrow C$: $\{M\}Kc$; $\{T\}Ks \subset M$
2. $C \rightarrow S$: $\{T\}Ks$,

где К - Kerberos;
С - клиент;
S - сервер;
Кх - ключ шифрования принадлежащий субъекту X;
{Y}Кх - информация Y зашифрована секретным ключом субъекта X.

Достоинства:

- Конфиденциально;
- Воспользоваться информацией может только целевой субъект;
- В открытом виде нигде не появляется.

Рассмотрим несколько вариантов протокола обмена при аутентификации [1,5,6]. При этом пока не рассматриваются следующие вопросы:

- первоначальный обмен ключами между Kerberos'ом и субъектами;
- как субъекты хранят свои ключи;
- как осуществляется администрирование Kerberos.

Первый вариант.

1. $C \rightarrow K$: c, s, ...
2. $K \rightarrow C$: $\{d1\}_{Kc}$; $\{T_{c\text{-}s}\}_{Ks}$
3. $C \rightarrow S$: d2, $\{T_{c\text{-}s}\}_{Ks}$

где $T_{c\text{-}s}$ - *билет клиента С к серверу S;*

d1 - некоторая информация, которую Kerberos передает клиенту и серверу (в билете к серверу);

d2 - часть информации из d1, ее сервер сравнивает с такой же, находящейся в билете.

Чтобы с помощью Kerberos получить доступ к серверу S, клиент С посылает запрос, содержащий сведения о себе и о требуемой услуге.

В ответ Kerberos возвращает две порции информации: так называемый билет, зашифрованный секретным ключом сервера, и копию части информации из билета, зашифрованную секретным ключом клиента.

Клиент должен расшифровать вторую порцию данных и переслать ее вместе с билетом серверу. Сервер, расшифровав билет, может сравнить его содержимое с дополнительной информацией, переданной клиентом.

Совпадение свидетельствует о том, что клиент смог расшифровать предназначенные ему данные - ведь содержимое билета никому, кроме сервера и Kerberos, недоступно – и продемонстрировал знание своего секретного ключа. Исходя из предпосылки считаем, что субъект тот, за кого себя выдает.

Это очень упрощенный вариант протокола. Он не обеспечивает полностью защищенный процесс аутентификации. Например, злоумышленник, перехватив диалог, может послать серверу ту же информацию и выдать себя за второго С. В более практичных вариантах процедур проверки аутентичности добавляют:

- разделяемый сеансовый ключ для обеспечения конфиденциальности взаимодействия;
- срок годности этого ключа, билета, по истечении которого необходимо заново пройти процедуру аутентификации (стандартный срок – 8 часов – может быть изменен в зависимости от политики безопасности данной организации);
- для подтверждения подлинности сервера S клиенту С – клиент шифрует передаваемую серверу дополнительную информацию с помощью сеансового ключа, который Kerberos поместил в билет, зашифрованный на секретном ключе сервера. Сервер, расшифровав дополнительную информацию клиента, возвращает часть ее клиенту (тоже соответственно зашифрованную на сеансовом ключе);
- для защиты от воспроизведения (выдачи злоумышленником себя в качестве второго клиента С):
- временной штамп – удостовериться в "свежести" сообщения;
- случайное число – неповторяемость сообщения и подтверждение целостности цепочки сообщений (помещая это число или закономерно модифицированное (например, +1)).

Второй вариант.

1. $C \rightarrow K$: {c, s1, ...}
2. $K \rightarrow C$: {n, s,timeexp $K_{c\text{-}s}$, ...}$_{Kc}$, {T}$_{Ks}$
3. $C \rightarrow S$: {s1, ts, ck ...}$_{Kc\text{-}s}$, {T}$_{Ks}$
4. $S \rightarrow C$: {ts, c, timeexp,...}$_{Kc\text{-}s}$,

где $K_{c\text{-}s}$ - разделяемый клиентом С и сервером S сеансовый ключ;
s1 - необходимый клиенту сервис;
n - одноразовое число;
timeexp - срок годности билета;
ts - временной штамп;
ck - контрольная сумма.

В билет помещается следующая информация:

имя С - сервер определяет клиента, запрашивающего определенный сервис;

имя самого сервиса – подтверждение правильной расшифровки и, соответственно, правильной генерации сообщения к серверу со стороны клиента, а также знания разделяемого сеансового ключа (сеансовый ключ помещается в ответ Kerberos'a клиенту и шифруется на секретном ключе клиента);

сетевой адрес – для частичной (сетевой адрес тоже можно подделать) защиты от кражи использования билета, а также для идентификации клиента;

Порции информации, позволяющие убедиться в подлинности предъявившего их субъекта, назовем аутентификатором (А).

Если клиенту понадобятся услуги нескольких серверов, то придется, соответственно, несколько раз доказывать свою подлинность, взаимодействовать, использовать секретный ключ. Для минимизации времени активного использования секретного ключа разделяют Kerberos на сервер начальной аутентификации (AS – Authentication Server) и сервер выдачи билетов (TGS – Ticket Granting Server).

Третий вариант.

1. $C \rightarrow AS$: c, tgs1, ...
2. $AS \rightarrow C$: {$K_{c\text{-}tgs}$, tgs, ...}$_{Kc}$, {Tc-tgs}$_{Ktgs}$
3. $C \rightarrow TGS$: {A_c}$_{Kc\text{-}tgs}$, {$T_{c\text{-}tgs}$}$_{Ktgs}$, s1,...
4. $TGS \rightarrow C$: {$K_{c\text{-}s}$, s,...}$_{Kc\text{-}tgs}$, {$T_{c\text{-}s}$}$_{Ks}$
5. $C \rightarrow S$: {Ac}$_{Kc\text{-}s}$, {$T_{c\text{-}s}$}$_{Ks}$
6. $S \rightarrow C$: {As}$_{Kc\text{-}s}$,

где AS - Authentication Server – сервер начальной аутентификации;
TGS - Ticket Granting Server - сервер выдачи билетов;
$T_{c\text{-}tgs}$ - билет клиента С к серверу TGS;
Ac - аутентификатор клиента С;
As - аутентификатор сервера S.

AS выдает TGT (Ticket Granting Ticket – "билет на билеты"), на основании которого у TGS получает билеты к другим серверам.

В том числе, при использовании глобальных сетей, субъекты из одной области управления могут взаимодействовать с субъектами из другой области управления. (Естественно, при наличии соглашения между областями управления и после обмена ключами между Kerberos'ами этих областей. Возможны как иерархические, так и специальные отношения и соглашения между областями управления).

В протоколе это отражается в разделении TGS на локальный locTGS и удаленный remTGS. При наличии договорных отношений локальный TGS выдает билеты к удаленному TGS, а тот – билеты к удаленным серверам.

Четвертый вариант.

1. $C \rightarrow$ locAS: c, tgs,...
2. locAS $\rightarrow C$: {$K_{c\text{-}tgs}$, tgs,...}$_{Kc}$, {$T_{c\text{-}tgs}$}$_{Ktgs}$

3. $C \rightarrow locTGS$: $\{A_c\}_{Kc\text{-}tgs}$, $\{T_{c\text{-}tgs}\}_{Ktgs}$, tgs-rem,...
4. $locTGS \rightarrow C$: $\{K_{c\text{-}remTGS}$, remTGS,...$\}_{Kc\text{-}tgs}$, $\{T_{c\text{-}remTGS}\}_{KremTGS}$
5. $C \rightarrow remTGS$: $\{A_c\}_{Kc\text{-}remTGS}$, $\{T_{c\text{-}remTGS}\}_{KremTGS}$, remS,...
6. $RemTGS \rightarrow C$: $\{K_{c\text{-}remS}$, remS,...$\}K_{c\text{-}remTGS}$, $\{T_{c\text{-}remS}\}_{KremS}$
7. $C \rightarrow remS$: $\{A_c\}_{Kc\text{-}remS}$, $\{T_{c\text{-}remS}\}_{KremS}$
8. $RemS \rightarrow C$: $\{A_{remS}\}_{Kc\text{-}remS}$

Помимо описанных возможностей существует множество дополнительных. Например, вводится срок годности – устанавливается не только окончание, но и начало срока (так называемые "билеты на потом"). Вводится возможность продления билета и максимальный срок такого продления; передача прав одного субъекта на действия от имени другого (сервер печати к файлам пользователя). Каждому билету приписываются разнообразные флаги.

Для реализации выбран третий вариант протокола. Он соединяет в себе такие противоположные качества, необходимые в экспериментальной версии, как простота и функциональность. Он самый простой из функциональных и самый функциональный из простых. Этот протокол сравнительно легко расширить до наиболее сложного известного протокола и внести пока не изобретенные усовершенствования.

Проанализируем возможность применения средств симметричного шифрования (СШ). В оригинальных разработках системы Kerberos на всех этапах применяется симметричное шифрование. После размышлений над этим фактом авторы пришли к выводу о том, что в данной ситуации (эта версия, этот протокол, опытный характер разработки) использование несимметричных систем криптопреобразований является неконструктивным. Увеличивается время преобразований, усложняется генерация ключей (например, удваивается база ключей).

Существующие реализации Kerberos основаны на использовании симметричных алгоритмов, принятых в качестве стандартных в странах-разработчиках. Для использования системы в отечественных средствах, необходима разработка системы на основе стандартов шифрования, принятых на Украине. Так как время жизни стандарта кратковременно, одним из важных требований к системе является возможность простой замены используемых алгоритмов, длин ключей, способов их хранения.

Кратко опишем результаты выполнения этапов анализа и проектирования.

Было принято решение произвести пробную реализацию сервиса Kerberos и выполнить ее в объектно-ориентированной парадигме (как проектирования, так и программирования).

В результате анализа проблемной области были выделены классы и составлена иерархия (упрощенное изображение иерархии уровня приложений показано на рисунке 1), которая, в зависимости от уровня абстракции, раскладывается на три плоскости:

- классы абстракций уровня приложений;
- классы абстракций уровня сообщений
- классы абстракций уровня базовых структур данных.

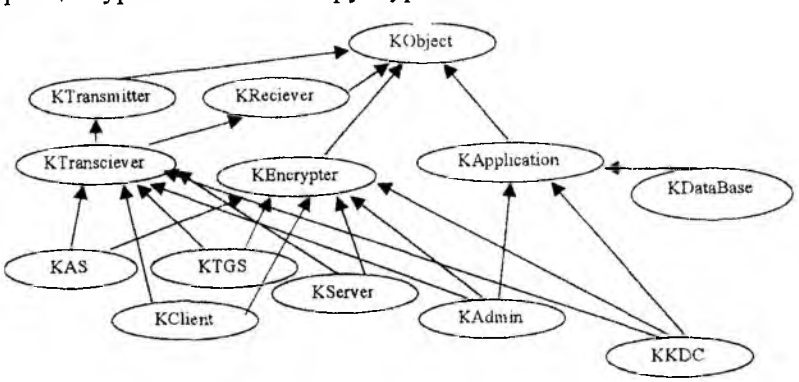

Рис. 1

На данный момент в иерархии 52 класса, что вынуждает более внимательно подойти к вопросам управления библиотекой классов [8,9]. При реализации мы придерживаемся следующих соглашений:

- класс ↔ файл – Установление однозначной связи. Повышает гибкость библиотеки и обеспечивает раздельную компиляцию.
- файл ↔ переменная препроцессора – Позволяет на этапе компиляции отслеживать какие файлы уже компилировались.
- условная компиляция – Использование препроцессорной обработки позволяет полностью исключить отладочный код из коммерческой версии.
- идентификация классов – каждый класс должен «уметь» сообщить о состоянии своего объекта, его выполненных действиях, размещении в памяти.
- расширяемость – необходимость явно спроектировать возможность расширения путем предоставления свободно модифицируемого каркаса [9].
- настраиваемость – необходимость предусматривать пути использования и приспособления данной библиотеки [9].

Вследствие постоянно увеличивающегося количества классов в иерархии автором в настоящее время рассматривается возможность использования CASE – средств (например, Rational Rose) [10,11].

Список литературы: **1.** *Вьюкова Н.* Сервер аутентификации Kerberos //Открытые системы. 1996. №1. с 44-50. **2.** *John Kohl and B. Clifford Neuman.* The Kerberos Network Authentication Service (Version 5). Internet Request for Comments RFC 1510. September 1993. **3.** *John Linn.* The Kerberos Version 5 GSS-API Mechanism. Internet Request for Comments RFC 1964. **4.** *B. Clifford Neuman and Theodore Ts'o. Kerberos:* An Authentication Service for Computer Networks, IEEE Communications, 32(9):33-38. September 1994. **5.** *Brian Tung.* The Moron's Guide to Kerberos. http://gost.isi.edu/brian/security/kerberos.html **6.** *Bill* Bryant. Designing an Authentication System: a Dialogue in Four Scenes. 1988. Afterword by Theodore Ts'o, 1997 http://web.mit.edu/kerberos/www/dialogue.html **7.** *S.M. Bellovin and M. Merritt.* Limitations of the Kerberos Authentication System. In *Proceedings of the Winter 1991 Usenix Conference*. January 1991. ftp://research.att.com/dist/internet_security/kerblimit.usenix.ps **8.** *Дьюхаст С., Старк К.* Программирование на C++. Пер с англ. Киев: «ДиаСофт», 1993. 272 с. **9.** *Страуструп Б.* Язык программирования C++, Часть вторая. Пер. с англ. Киев: «ДиаСофт», 1993. 296 с. **10.** *Вендоров А.М.* Современные методы и средства проектирования информационных систем. Москва: Финансы и статистика, 1998. 176 с. **11.** *Калянов Г.Н.* Консалтинг при автоматизации предприятий: Научно-практическое издание. Серия «Информатизация России на пороге XXI века». Москва: СИНТЕГ, 1997. 316 с.

Харьковский государственный технический университет радиоэлектроники

Поступила в редколлегию 15.03.2000