Network Working Group
Request for Comments: 2222
Category: Standards Track

J. Myers
Netscape Communications

October 1997

Простой уровень аутентификации и защиты SASL

Simple Authentication and Security Layer (SASL)

Статус документа

В этом документе содержится спецификация протокола, предложенного сообществу Internet. Документ служит приглашением к дискуссии в целях развития и совершенствования протокола. Текущее состояние стандартизации протокола вы можете узнать из документа Internet Official Protocol Standards (STD 1). Документ может распространяться без ограничений.

Авторские права

Оглавление

1. Аннотация

В этом документе определён метод добавления поддержки аутентификации для основанных на соединениях протоколов. Для использования данной спецификации протокол включает команду идентификации и аутентификации[1] пользователя на сервере, а также для дополнительного согласования защиты последующих транзакций протокола. Если использование команды согласовано, между протоколом и соединением помещается уровень защиты. В данном документе содержится спецификация этой команды, определяется несколько механизмов для использования этой командой и протокол, используемый для передачи информации в процессе согласования уровня защиты через соединение.

2. Организация документа

2.1. Как работать с этим документом

Этот документ адресован двум разным категориям читателей - разработчикам протоколов, которые будут использовать эту спецификацию для поддержки аутентификации в своих протоколах, и разработчикам клиентских и серверных приложений для протоколов, использующих данную спецификацию.

[1] Проверки подлинности. *Прим. перев.*

Главы "Введение и обзор", "Профильные требования" и "Вопросы безопасности" включают обсуждение вопросов, которые разработчики должны понимать, Эти главы важны при создании вариантов спецификации для использования с конкретными протоколами.

Разработчикам протоколов, использующих эту спецификацию потребуется также связанная с протоколом профильная информация в дополнение к информации, содержащейся в этом документе.

2.2. Используемые в документе соглашения

В примерах обозначения "C:" и "S:" показывают строки, передаваемые клиентом и сервером, соответственно.

Ключевые слова необходимо (MUST), недопустимо (MUST NOT), следует (SHOULD), не следует (SHOULD NOT), возможно (MAY) в данном документе должны интерпретироваться в соответствии с документом Key words for use in RFCs to Indicate Requirement Levels [RFC2119].

2.3. Примеры

Приведённые в документе примеры относятся к варианту этой спецификации для протокола IMAP [RFC 2060]. Кодирование base64 для запросов и откликов и включение префикса "+ " в отклики являются частью варианта IMAP4, а не самой спецификации SASL.

3. Введение и обзор

SASL[1] представляет собой метод добавления поддержки аутентификации для основанных на соединениях протоколов. Для использования данной спецификации протокол включает команду идентификации и аутентификации пользователя на сервере, а также для дополнительного согласования защиты последующих транзакций протокола.

Команда использует обязательный аргумент, идентифицирующий механизм SASL. Механизмы SASL идентифицируются именами длиной от 1 до 20 символов, включающими заглавные буквы[2], цифры, а также знаки дефиса (-) и подчёркивания (_). Имена механизмов SASL должны регистрироваться в IANA. Процедуры регистрации новых механизмов SASL описаны в главе "Требования к вариантам".

Если сервер поддерживает запрошенный механизм, он инициирует обмен данными аутентификации. Этот обмен включает последовательность запросов (challenge) сервера и откликов клиента, определяемых механизмом аутентификации. Запросы и отклики определяются механизмом аутентификации как бинарные маркеры (token) произвольной длины. Профиль протокола указывает способы кодирования маркеров и передачи их через соединение.

После получения команды аутентификации или любого отклика клиента сервер может выдавать запрос и индицировать отказ или успешное завершение аутентификации. Профиль протокола указывает способы индикации.

После получения запроса клиент может вернуть отклик на него или прервать обмен. Профиль протокола задаёт способы выполнения этих операций.

В процессе обмена аутентификационными данными механизм выполняет аутентификацию, передаёт серверу от клиента “идентификацию личности” (часто называется идентификатором пользователя - userid) и согласует используемый уровень защиты. Если согласовано использование уровня защиты, механизм должен также определить и согласовать максимальный размер буфера шифрованного текста, который каждая из сторон способна принять.

Передаваемая идентификация пользователя может отличаться от идентификации, используемой в процессе аутентификации клиента. Это позволяет агентам (таким, как proxy-серверы) выполнять процедуру аутентификации с использованием своих свидетельств, запрашивая полномочия “личности” для которой агент выполняет посреднические (proxy) функции. При использовании любого механизма передача пустой идентификации говорит серверу, что следует идентифицировать клиента по его аутентификационному свидетельству.

Если согласовано использование уровня защиты, этот уровень применяется для всех данных, передаваемых впоследствии через это соединение. Использование уровня защиты начинается сразу же вслед за последним откликом аутентификационного обмена для данных, отправленных клиентом, и индикацией завершения для данных, отправленных сервером. После вступления в силу уровня защиты поток данных протокола обрабатывается уровнем защиты в буферах шифрованного текста. Каждый буфер передаётся через соединение как поток октетов, которому предшествует 4-октетное поле с сетевым порядком байтов, показывающее размер следующего буфера. Размер буфера шифрованного текста не может превышать максимальный размер, определённый или согласованный другой стороной.

4. Профильные требования

Для использования данной спецификации определение протокола должно обеспечивать следующие данные:

1. Имя сервиса, которое будет выбираться из реестра IANA для элементов service в формате GSSAPI [RFC 2078].
2. Определение команды инициирования аутентификационного обмена. Эта команда должна включать в качестве параметра имя механизма, который будет выбирать клиент.

   Команде следует поддерживать дополнительный параметр, задающий изначальный отклик. Этот необязательный параметр позволяет клиенту избавиться от необходимости кругового обхода при использовании механизмов, в которых клиент первым передаёт данные. Когда клиент передаёт изначальный отклик, а выбранный механизм задаёт начало обмена с передачи запроса сервера, команда возвращает отказ. Дополнительную информацию по этому вопросу вы найдёте в параграфе 5.1.
3. Определение метода аутентификационного обмена, включающее кодирование запросов и откликов, индикацию завершения и отказа, способы прерывания обмена клиентом и взаимодействие метода обмена с присущими протоколу ограничениями размера.

[1] Simple Authentication and Security Layer - простой уровень аутентификации и защиты.
[2] Латиница. *Прим. перев.*

4. Идентификация октетов, с которых начинается работа уровня защиты, для обоих направлений.

5. Способ интерпретации свидетельства, переданного серверу от клиента.

5. Частные вопросы

5.1. Клиент должен начинать аутентификационный обмен

Некоторые механизмы задают для клиента необходимость первому передавать данные в процессе аутентификационного обмена.

Если профиль протокола поддерживает команду, которая инициирует аутентификационный обмен путём передачи изначального отклика клиента, следует использовать параметр этой команды.

Если параметр с изначальным откликом клиента не задан или профиль протокола не поддерживает команду, позволяющую задать этот параметр, сервер передаёт пустой запрос. Отклик клиента на этот запрос будет использоваться в качестве начального (после этого сервер передаёт следующий запрос и индикацию отказа или успешного завершения[1]).

5.2. Сервер возвращает сообщение об успехе с дополнительными данными

Некоторые реализации могут использовать передачу дополнительной информации в сообщении (challenge) сервера об успешном завершении обмена. Эти данные могут, например, аутентифицировать сервер для клиента.

Если профиль протокола не разрешает серверу передавать сообщение об успешном завершении обмена с дополнительными данными, сервер будет передавать эти данные в виде отдельного запроса (challenge) без индикации завершения обмена. Клиент возвращает пустой отклик в ответ на такой запрос и после получения этого отклика сервер передает индикацию успешного завершения обмена.

5.3. Множественная аутентификация

Если иное явно не указано в профиле протокола, допускается только одно успешное согласование SASL в данной сессии. В таких случаях после успешного завершения аутентификационного обмена все последующие попытки инициировать такой обмен в той же сессии завершаются отказом.

В тех случаях, когда профиль явно позволяет множество успешных согласований SASL ни в коем случае не может одновременно использоваться несколько уровней защиты. Если уровень защиты уже активизирован и следующее согласование SASL не устанавливает уровень защиты, прежний уровень продолжает работать. Если же при следующем согласовании SASL задан другой уровень защиты, этот уровень будет использоваться взамен предыдущего.

6. Процедуры регистрации

Регистрация механизма SASL производится путём заполнения формы, приведённой в параграфе 6.4, и отправки ее по адресу iana@isi.edu. Агентство IANA имеет право отвергать явно ложные регистрации и не будет рецензировать заявки.

Для имён механизмов SASL не используется каких-либо соглашений об именовании - можно регистрировать любое имя, соответствующее синтаксису механизма SASL.

Хотя процедуры регистрации и не требуют этого, авторам механизмов SASL рекомендуется просмотреть отклики сообщества и комментарии. Для получения откликов сообщества авторы могут опубликовать свои предложения как черновой вариант (internet-draft). Механизмы SASL, предназначенные для широкого использования, следует стандартизовать с использованием обычных процедур IETF, когда это возможно.

6.1. Регистрация комментариев к механизмам SASL

Комментарии по поводу зарегистрированных механизмов SASL следует направлять сначала “владельцу” соответствующего механизма. Авторы комментариев могут (после разумного числа попыток контакта с владельцем) передать в IANA запрос на включение своих комментариев в регистрацию соответствующего механизма SASL. После одобрения IANA эти комментарии станут доступными вместе с механизмом SASL.

6.2. Список зарегистрированных механизмов SASL

Список зарегистрированных механизмов SASL доступен анонимным пользователям по протоколу FTP на сайте ftp://ftp.isi.edu/in-notes/iana/assignments/sasl-mechanisms/, эти механизмы также включаются в периодически обновляемый документ "Assigned Numbers" [в настоящее время STD 2, RFC 1700[2]]. Описания механизмов SASL и другие материалы, связанные с их поддержкой могут также публиковаться как Informational RFC путём их отправки по адресу rfc-editor@isi.edu (документ должен соответствовать инструкциям для авторов RFC [RFC 2223]).

6.3. Контроль изменений

После публикации механизма SASL, зарегистрированного IANA, автор может подать запрос на изменение этого механизма. Процедура изменения выполняется так же, как регистрация механизма.

Владелец механизма SASL может передать ответственность за этот механизм другому лицу или организации, проинформировав об этом агентство IANA; такая передача не требует обсуждения или рассмотрения.

IESG может передать другому лицу ответственность за механизм SASL. Обычно это делается в целях обеспечения возможности изменения механизма после смерти автора, утраты контактов с ним или по иным причинам, которые делают невозможным внесение важных для сообщества изменений.

---

[1] В зависимости от полученных откликов на запросы. *Прим. перев.*

[2] В соответствии с RFC 3232 этот документ утратил силу. *Прим. перев.*

Регистрация механизма SASL не может быть отменена (удалена) - механизмы, которые представляются неприемлемыми для дальнейшего использования, могут быть объявлены вышедшими из употребления (OBSOLETE) путём изменения поля "intended use" в регистрационной форме; этот факт явно указывается в публикуемых IANA списках.

IESG рассматривается как владелец всех механизмов SASL, которые опубликованы со статусом IETF standards track.

6.4. Регистрационная форма

To: iana@iana.org

Subject: Registration of SASL mechanism X

SASL mechanism name:

Security considerations:

Published specification (optional, recommended):

Person & email address to contact for further information:

Intended usage:

(Один из вариантов - COMMON, LIMITED USE или OBSOLETE)

Author/Change controller:

(любая информация, которую автор сочтет интересной, может быть добавлена после этой строки).

7. Определения механизмов

В данной главе рассмотрены определенные к настоящему моменту механизмы SASL.

7.1. Механизм Kerberos версии 4

Имя механизма, связанного с Kerberos версии 4, - "KERBEROS_V4".

Первый запрос (challenge) представляет собой 32-битовое случайное значение с сетевым порядком байтов. Клиент отвечает квитанцией[1] Kerberos и свидетельством[2] для доверителя[3] "service.hostname@realm", где service - имя сервиса, указанное в профиле протокола, hostname - первая компонента имени хоста для сервера с использованием только строчных букв, а realm - Kerberos realm для сервера. Шифрованное поле контрольной суммы, включённое в свидетельство Kerberos, содержит запрос (challenge) сервера в сетевом порядке байтов.

После расшифровки и проверки квитанции и свидетельства сервер проверяет совпадение значения поля контрольной суммы с переданным ранее 32-битовым случайным числом. Если значения совпадают, сервер должен добавить 1 к контрольной сумме и создать 8 октетов данных, из которых первые четыре октета содержат увеличенную на 1 контрольную сумму с сетевым порядком байтов, пятый октет содержит битовую маску, задающую уровни безопасности, которые поддерживает сервер, три оставшиеся октета задают (сетевой порядок байтов) максимальный размер буфера шифрованного текста, который сервер способен принять. Сервер должен зашифровать с использованием режима DES ECB эти 8 октетов данных в сеансовый ключ и ввести зашифрованные данные во второй запрос. Клиент принимает аутентификацию сервера, если первые 4 октета расшифрованных данных представляют собой число, на единицу превышающее полученное ранее значение контрольной суммы.

Клиент должен создать данные с первыми 4 октетами, содержащими исходную контрольную сумму, полученную от сервера, с сетевым порядком байтов. Пятый октет должен содержать битовую маску, задающую выбранный уровень защиты, а октеты 6-8 - максимальный размер (сетевой порядок байтов) буфера шифрованного текста, который клиент способен принять. Остальные октеты содержат “идентификацию личности”. После этого клиент должен добавить в конце блока данных от 1 до 8 нулевых октетов для выравнивания данных по 8-октетной границе. Полученные данные клиент должен зашифровать, используя режим DES PCBC, с сеансовым ключом и передать зашифрованные данные в качестве отклика. Сервер должен удостовериться, что доверитель, идентифицируемый квитанцией Kerberos, уполномочен на организацию соединения как данная личность. После такой верификации процесс аутентификации считается завершённым.

Поддерживаются следующие уровни защиты и соответствующие битовые маски:

1 - нет защиты;

2 - защита целостности (krb_mk_safe);

4 - сохранение тайны (krb_mk_priv).

В будущем могут быть определены другие битовые маски, для непонятных битов должно предполагаться нулевое значение.

Ниже показаны два сценария регистрации (login) с использованием Kerberos версии 4 для протокола IMAP4 (отметим, что длинные строки разбиты на несколько строк лишь для наглядности)

```
S: * OK IMAP4 Server
C: A001 AUTHENTICATE KERBEROS_V4
S: + AmFYig==
C: BAcAQU5EUkVXLkNNVS5FRFUAOCAsho84kLN3/IJmrMG+25a4DT+nZImJjnTNHJUtxAA+o0KPKfHEcAFs9a3CL5Oe
   be/ydHJUwYFdWwuQ1MWiy6IesKvjL5rL9WjXUb9MwT9bpObYLGOKi1Qh
S: + or//EoAADZI=
C: DiAF5A4gA+oOIALuBkAAmw==
```

[1]Ticket.
[2]Authenticator.
[3]Principal.

```
S: A001 OK Kerberos V4 authentication successful
S: * OK IMAP4 Server
C: A001 AUTHENTICATE KERBEROS_V4
S: + gcfgCA==
C: BAcAQU5EUkVXLkNNVS5FRFUAOCAsho84kLN3/IJmrMG+25a4DT+nZImJjnTNHJUtxAA+o0KPKfHEcAFs9a3CL5Oe
   be/ydHJUwYFdWwuQ1MWiy6IesKvjL5rL9WjXUb9MwT9bpObYLGOKi1Qh
S: A001 NO Kerberos V4 authentication failed
```

7.2. Механизм GSSAPI

Имя связанное со всеми механизмами, использующими GSSAPI [RFC 2078] - GSSAPI.

7.2.1 Клиентская сторона аутентификационного обмена

Клиент вызывает GSS_Init_sec_context, передавая 0 для input_context_handle (изначально) и targ_name = output_name из GSS_Import_Name, вызванной с input_name_type = GSS_C_NT_HOSTBASED_SERVICE и input_name_string = "service@hostname", где "service" - имя службы, заданное в профиле протокола, а "hostname" полное имя серверного хоста. Далее клиент передаёт отклик с полученным output_token. Если GSS_Init_sec_context возвращает GSS_S_CONTINUE_NEEDED, клиенту следует ожидать что сервер введёт маркер в последующем запросе. Клиент должен передать маркер другому вызову GSS_Init_sec_context, повторяя описанные выше операции.

Когда GSS_Init_sec_context возвращает GSS_S_COMPLETE, клиент предпринимает перечисленные ниже действия. Если при последнем вызове GSS_Init_sec_context было возвращено значение output_token, клиент передаёт отклик с output_token, в противном случае передаётся отклик без данных. Клиенту после этого следует ожидать, что сервер введёт маркер в последующем запросе. Клиент передаёт этот маркер GSS_Unwrap и интерпретирует первый октет полученного в результате открытого текста, как битовую маску, задающую уровни защиты, которые поддерживает сервер, а октеты 2-4 - как максимальный размер output_message для передачи серверу. После этого клиент создаёт данные, первый октет которых содержит битовую маску выбранного уровня защиты, октеты 2-4 (сетевой порядок байтов) - максимальный размер output_message, который клиент способен принять, а остающиеся октеты содержат свидетельство[1]. Клиент передаёт данные GSS_Wrap с conf_flag = FALSE и отвечает серверу полученным output_message. После этого клиент считает аутентификацию сервера завершенной.

7.2.2 Серверная сторона аутентификационного обмена

Сервер передаёт изначальный запрос клиента функции GSS_Accept_sec_context как input_token, устанавливая input_context_handle = 0 (изначально). Если GSS_Accept_sec_context возвращает GSS_S_CONTINUE_NEEDED, сервер возвращает клиенту в запросе сгенерированный маркер output_token и передаёт результирующий отклик другому вызову GSS_Accept_sec_context, повторяя описанные выше действия.

Когда GSS_Accept_sec_context возвращает GSS_S_COMPLETE, сервер[2] выполняет перечисленные ниже операции. Если при последнем вызове GSS_Accept_sec_context был возвращён output_token, сервер возвращает этот маркер клиенту в запросе и ожидает от клиента отклика, не содержащего данных. Независимо от того, был ли возвращён маркер output_token (и после получения от клиента любого отклика на такой output_token), сервер создаёт 4 октета данных, из которых первый октет представляет собой битовую маску поддерживаемых сервером уровней защиты, а октеты 2-4 (сетевой порядок байтов) - максимальный размер output_token, который сервер способен принять. После этого сервер должен передать открытый текст функции GSS_Wrap с conf_flag = FALSE и ввести сгенерированное сообщение output_message в запрос клиенту. После этого сервер должен передать полученный в результате отклик функции GSS_Unwrap и интерпретировать первый октет полученного открытого текста как битовую маску выбранного клиентом уровня защиты. Октеты 2-4 (сетевой порядок байтов) трактуются как максимальный размер output_message для передачи клиенту, а остальные октеты - как свидетельство. Сервер должен убедиться, что src_name имеет полномочия на аутентификацию как идентифицированная “личность”. После такой верификации процесс аутентификации завершается.

7.2.3 Уровень защиты

Уровни защиты и соответствующие им битовые маски показаны ниже:

1 - нет защиты

2 - защита целостности.

Отправитель вызывает GSS_Wrap с conf_flag = FALSE

4 - сохранение тайны.

Отправитель вызывает GSS_Wrap с conf_flag = TRUE

В будущем могут быть определены другие битовые маски; непонятные биты должны трактоваться как имеющие нулевое значение.

7.3. Механизм S/Key

Имя, связанное с механизмом S/Key [RFC 1760], использующим MD4, - "SKEY".

Клиент передаёт изначальный запрос со свидетельством.

Сервер после этого вводит запрос, содержащий десятичный порядковый номер, за которым следует один пробел и “строка затравки[3]” для показанного свидетельства. Клиент отвечает одноразовым паролем, заданным 64-битовым значением с сетевым порядком байтов или закодированным в формат "six English words".

Сервер должен проверить одноразовый пароль. После такой проверки процесс аутентификации завершается.

---

[1] Authorization identity.
[2] В оригинальном документе ошибочно указан клиент. *Прим. перев.*
[3] Seed string.

Аутентификация S/Key не поддерживает никаких уровней защиты.

Пример:Ниже показаны два сценария регистрации (login) S/Key для протокола IMAP4.

```
S: * OK IMAP4 Server
C: A001 AUTHENTICATE SKEY
S: +
C: bW9yZ2Fu
S: + OTUgUWE1ODMwOA==
C: Rk9VUiBNQU5OIFNPT04gRklSIFZBUlkgTUFTSA==
S: A001 OK S/Key authentication successful
S: * OK IMAP4 Server
C: A001 AUTHENTICATE SKEY
S: +
C: c21pdGg=
S: + OTUgUWE1ODMwOA==
C: BsAY3g4gBNo=
S: A001 NO S/Key authentication failed
```

Приведенный ниже сценарий S/Key-регистрации для протокола типа IMAP4, который использует дополнительный аргумент “изначального отклика в команде AUTHENTICATE.

```
S: * OK IMAP4-Like Server
C: A001 AUTHENTICATE SKEY bW9yZ2Fu
S: + OTUgUWE1ODMwOA==
C: Rk9VUiBNQU5OIFNPT04gRklSIFZBUlkgTUFTSA==
S: A001 OK S/Key authentication successful
```

7.4. Внешний механизм

Имя механизма, связанное с внешней аутентификацией, - "EXTERNAL".

Клиент передаёт изначальный запрос со свидетельством.

Сервер использует эту информацию за пределами SASL для того, чтобы убедиться в наличии у клиента полномочий для аутентификации с этим свидетельством. Если клиент имеет такие полномочия, сервер показывает успешное завершение аутентификационного обмена, в противном случае сервер индицирует отказ.

Системой, обеспечивающей внешнюю информацию может быть, например, IPsec или TLS.

Если клиент шлёт в качестве свидетельства пустую строку (показывая таким образом, что идентификация производится на основе представленных клиентом аутентификационных данных[2]), клиент идентифицируется на основе аутентификационных данных, которые имеются в системе, обеспечивающей внешнюю аутентификацию.

8. Литература

[RFC 2060] Crispin, M., "Internet Message Access Protocol - Version 4rev1", RFC 2060, December 1996.

[RFC 2078] Linn, J., "Generic Security Service Application Program Interface, Version 2", RFC 2078, January 1997.

[RFC 2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, March 1997.

[RFC 2223] Postel, J., and J. Reynolds, "Instructions to RFC Authors", RFC 2223, October 1997.

[RFC 1760] Haller, N., "The S/Key One-Time Password System", RFC 1760, February 1995.

[RFC 1700] Reynolds, J., and J. Postel, "Assigned Numbers", STD 2, RFC 1700[3], October 1994.

9. Вопросы безопасности

В этом документе рассматриваются вопросы безопасности.

Механизмы, поддерживающие защиту целостности, устроены так, что обеспечивается целостность данных при согласовании уровня защиты и передаче удостоверяющей информации. Когда клиент выбирает уровень защиты, поддерживающий по крайней мере целостность данных, ему обеспечивается защита от активных атак с захватом соединений и изменения аутентификационных данных в процессе согласования соединения.

Когда сервер или клиент поддерживает множество механизмов аутентификации, отличающихся уровнем обеспечиваемой защиты, при активной атаке могут быть предприняты попытки принудить к использованию наименее защищённого механизма. Для защиты от таких атак клиенту или серверу, поддерживающему несколько механизмов с различным уровнем защиты, следует задавать в конфигурационных параметрах минимальный уровень, который может использоваться для соединений. Установки минимального уровня только на сервере недостаточно, поскольку атакующий может изменить механизм, показываемый клиенту в качестве поддерживаемого, заставляя тем самым клиента использовать аутентификационные данные для наименее защищённого механизма.

Клиентский выбор механизма SASL указывается явно и может быть подменен в результате активной атаки. Для всех разрабатываемых механизмов SASL важно обеспечить предотвращение аутентификации атакующего при использовании наиболее низкого уровня защиты в результате подмены имени механизма и/или запросов и откликов.

Все протокольные транзакции до завершения процесса аутентификации производятся в открытом виде и могут быть изменены в результате активной атаки. В тех случаях, когда клиент выбирает защиту целостности, важно обеспечить согласование всех важных с точки зрения безопасности параметров только после успешного завершения процедуры аутентификации. Протоколам следует игнорировать результаты согласований, выполненных до завершения аутентификации или повторять согласование после того, как аутентификация будет завершена.

---

[2]Authentication credentials.

[3]В соответствии с RFC 3232 этот документе утратил силу. Информация о выделенных значения хранится в базе данных на сайте http://www.iana.org/numbers.html. *Прим. перев.*

10. Адрес автора

John G. Myers

Netscape Communications

501 E. Middlefield Road

Mail Stop MV-029

Mountain View, CA 94043-4042

EMail: jgmyers@netscape.com

Перевод на русский язык

Николай Малых

nmalykh@protokols.ru

Приложение A. Связь SASL с защитой на транспортном уровне

Возникают вопросы о связи SASL с различными механизмами (такими, как IPsec и TLS) организации защищённых соединений.

Ключевыми элементами SASL являются:

1. Разделение идентификации при проверке полномочий (authorization identity) и идентификационных данных в предъявленных клиентом свидетельствах. Это позволяет агентам (таким, как proxy-серверы) выполнять процедуру аутентификации с использованием своих свидетельств, запрашивая полномочия сущности, для которой агент выполняет посреднические (proxy) функции.
2. После завершения аутентификационного обмена сервер знает, какого уполномоченного агента[1] клиент хочет использовать. Это позволяет серверу перейти в состояние протокола "user is authenticated[2]".

Эти функции весьма важны для некоторых прикладных протоколов, тогда как службы Transport Security[3] не обеспечивают их. Определение механизмов SASL на основе таких служб является неблагодарной задачей, поскольку основа этих служб является избыточной при наличии SASL, а некоторые методы, обеспечивающие важные функции SASL, придётся разрабатывать заново.

Иногда бывает важно разрешить для существующего соединения использование защищённых служб, которые отсутствуют в модели SASL (TLS является примером такой службы). Это можно сделать путём добавления в протокол соответствующей команды (например, "STARTTLS"). Такие команды выходят за пределы SASL и их следует отличать от команды, инициирующей аутентификационный обмен SASL.

В некоторых ситуациях разумно использовать SASL ниже одной из таких служб Transport Security. Транспортный сервис будет обеспечивать защиту соединения и аутентифицировать клиента, а SASL будет согласовывать уполномоченную личность. Согласование SASL будет переводить протокол из состояния unauthenticated в состояние authenticated. Механизм EXTERNAL в SASL явно предназначен для случаев, когда транспортный сервис защищает соединение и аутентифицирует клиента, а SASL согласует.

При использовании SASL ниже достаточно сильного сервиса Transport Security очевидно, что уровень защиты SASL становится избыточным. В таких случаях клиент и сервер могут согласовать отказ от использования уровня защиты SASL.

Полное заявление авторских прав

[1] Authorization identity.
[2] Подлинность пользователя засвидетельствована.
[3] Защита на транспортном уровне.