И. И. Зензин

ОБЕСПЕЧЕНИЕ БЕЗОПАСНОСТИ ИНФОРМАЦИИ В BITTORRENT-СЕТЯХ НА ОСНОВЕ ПРИМЕНЕНИЯ АЛГОРИТМА ЗАЩИТЫ ОТ ФАЛЬСИФИКАЦИИ УЧЕТНЫХ И СТАТИСТИЧЕСКИХ ДАННЫХ

Введение

Как известно, учетные данные пользователей и статистическая информация об объемах переданных в коллективное пользование и полученных информационных ресурсов в ходе их передачи серверу статистического учета в распределенной сети могут быть несанкционированно модифицированы. Несанкционированная модификация может произойти, в том числе, в процессе эксплуатации нарушителем таких уязвимостей, как подмена доверенного объекта распределенной сети, внедрение в распределенную сеть фиктивного объекта путем навязывания ложного маршрута. Для реализации механизма защиты учетных и статистических данных необходимо разработать алгоритм, который позволит предотвратить подмену и/или перехват учетных и статистических данных пользователей РВС.

1. Формирование исходных данных

Для решения поставленной задачи обозначим ряд требований, которым должен удовлетворять разрабатываемый алгоритм:

- Логин и пароль пользователя должен передаваться трекеру в шифрованном виде.
- Необходимо исключить возможность подмены и/или модификации передаваемого трекеру значения соотношения показателей предоставления и потребления ресурсов.
- Процесс создания и публикации файла метаданных должен включать в себя механизм подтверждения авторства и обеспечения целостности.

Предварительно необходимо представить информацию о принципе работы BitTorrent-протокола. Он заключается в следующем: нагрузка на распространителя файла уменьшается благодаря тому, что клиенты начинают обмениваться данными сразу же, даже если файл не получен ими до конца. Перед началом скачивания клиент подсоединяется к трекеру по адресу, указанному в торрент-файле, сообщает ему свой адрес и хеш-сумму торрент-файла, на что в ответ клиент получает адреса других клиентов, скачивающих или раздающих этот же файл. Далее клиент периодически информирует трекер о ходе процесса и получает обновленный список адресов. Этот процесс называется объявлением (*англ.* announce).

Клиенты соединяются друг с другом и обмениваются сегментами файлов без непосредственного участия трекера, который лишь хранит информацию, полученную от подключенных к обмену клиентов, список самих клиентов и статистическую информацию. Для эффективной работы сети BitTorrent необходимо, чтобы как можно больше клиентов были способны принимать входящие соединения. Неправильная настройка NAT или брандмауэра может этому помешать [1].

При соединении клиенты сразу обмениваются информацией об имеющихся у них сегментах. Клиент, желающий скачать сегмент (личер), посылает запрос и, если второй клиент готов отдавать, получает этот сегмент. После этого клиент проверяет контрольную сумму сегмента. Если она совпала с той, что записана в торрент-файле, то сегмент считается успешно скаченным и клиент оповещает всех присоединенных пиров о наличии у него этого сегмента. Если же контрольные суммы различаются, то сегмент начинает скачиваться заново. Некоторые клиенты блокируют тех пиров, которые слишком часто предоставляют некорректные сегменты.

Таким образом, объем служебной информации (размер торрент-файла и размер сообщений со списком сегментов) напрямую зависит от количества, а значит, и размера сегментов. Поэтому

при выборе сегмента необходимо соблюдать баланс: с одной стороны, при большом размере сегмента объем служебной информации будет меньше, но в случае ошибки проверки контрольной суммы придется скачивать еще раз больше информации. С другой стороны, при малом размере ошибки не так критичны, так как необходимо заново скачать меньший объем, но зато размер торрент-файла и сообщений об имеющихся сегментах становится больше.

Для написания торрент-файлов используется BEncode. Стоит добавить, что если массив байт может быть интерпретирован как строка (имена элементов в ассоциативном массиве, просто строковые поля), то используется кодировка UTF-8. Содержимое является одним большим ассоциативным массивом со следующими полями [2]:

info — вложенный ассоциативный массив, который, собственно, и описывает файлы, которые передает торрент;

announce — URL для трекера. Наряду с info является обязательным полем, остальные поля опциональны;

announce-list — список трекеров, если их несколько. В BEncode-виде — список списков;

creation date — дата создания. UNIX Timestamp;

comment — текстовое описание торрента;

created by — говорит нам о том, кем создан данный торрент.

Необходимо упомянуть то, что файлы представлены в протоколе частями. То есть файлы, содержащиеся в торренте, объединены в единый массив, и затем этот массив разделили на относительно небольшие части. В таком виде данные обрабатывает BitTorrent-протокол.

Ассоциативный массив info состоит из [3]:

piece length — размер одной части — 512 килобайт, 1 мегабайт и более. Слишком большое количество частей будет «раздувать» .torrent-файл;

pieces — строка, которая содержит конкатенацию SHA1-хешей, описывающих каждую часть. Длина этой строки равна 20 * количество частей;

name — рекомендательное имя файла (если файл один) или директории;

length — если файл один, то будет задано это поле, которое содержит длину файла;

files — если файлов несколько, то появится список ассоциативных массивов.

Формат элементов списка files:

length — длина файла;

path — список из строк, которые задают путь. Каждая строка — элемент пути, относительно корневой директории торрента. Для пути *a/b/c/d.jpg* будет 4 строки в данном списке — ['a', 'b', 'c', 'd.jpg'].

Также необходимо понимать процесс функционирования алгоритма обмена данными BitTorrent-протокола. Каждый клиент имеет возможность временно блокировать отдачу другому клиенту (*англ.* choke). Это делается для более эффективного использования канала отдачи. Кроме того, при выборе, кого разблокировать, предпочтение отдается пирам, которые сами передали этому клиенту много сегментов. Таким образом, пиры с хорошими скоростями отдачи поощряют друг друга по принципу «ты — мне, я — тебе».

Обмен сегментами ведется по принципу «ты — мне, я — тебе» симметрично в двух направлениях. Клиенты сообщают друг другу об имеющихся у них сегментах при подключении и затем при получении новых сегментов, и поэтому каждый клиент может хранить информацию о том, какие сегменты есть у других подключенных пиров. Порядок обмена выбирается так, чтобы сначала клиенты обменивались наиболее редкими сегментами: таким образом повышается доступность файлов в раздаче. В то же время выбор сегмента среди наиболее редких случаен, и поэтому можно избежать ситуации, когда все клиенты начинают скачивать один и тот же самый редкий сегмент, что негативно отразилось бы на производительности.

Обмен данными начинается, когда обе стороны в нем заинтересованы, то есть каждая из сторон имеет сегменты, которых нет у другой. Количество переданных сегментов подсчитывается, и, если одна из сторон обнаруживает, что передает в среднем больше, чем принимает, она блокирует на некоторое время отдачу другой стороне. Таким образом, в протокол заложена защита от недобросовестных потребителей.

Сегменты делятся на блоки размером 16-4096 килобайт, и каждый клиент запрашивает именно эти блоки. Одновременно могут запрашиваться блоки из разных сегментов. Более того, некоторые клиенты поддерживают скачивание блоков одного сегмента у разных пиров. В этом случае описанные выше алгоритмы и механизмы обмена применимы и к уровню блоков.

2. Алгоритм защиты учетных и статистических данных

Обозначим следующих основных участников процесса функционирования алгоритма:

- пользователь A;
- трекер;
- УЦ.

В роли пользователя A выступает любое лицо, принимающее участие в кооперативном обмене файлами и вычислительными заданиями. Трекер представляет собой сервер, на котором осуществляется публикация файлов метаданных, а также ведется учет статистических данных показателей предоставления и потребления ресурсов коллективного пользования. УЦ является комплексом средств, необходимых для корректного взаимодействия компонентов разрабатываемого алгоритма в части использования ЭП.

Для упрощения процесса разработки алгоритма введем следующие обозначения:

L — логин пользователя;

P — пароль пользователя;

$G_i вх(Q,T)$ — объем информационных ресурсов, полученных i-м пользователем в собственное пользование;

$G_i исх(Q,T)$— объем информационных ресурсов, предоставленных i-м пользователем в кооперативное пользование;

f_1 — разность между $G_i исх(Q,T)$ и $G_i вх(Q,T)$;

f_2 — $\frac{G_i вх(Q,T) + G_i исх(Q,T)}{2}$;

f_0 — f_1 / f_2;

Enc_K (X,Y) — шифрование пары (X,Y) на ключе K;

M1 — сообщение «Необходимо передать в кооперативное пользование ($G_i вх(Q,T)$ - $G_i исх(Q,T)$)-единиц трафика»;

M2 — сообщение «Соотношение f_0 достаточно для получения ресурсов в кооперативное пользование».

Далее рассмотрим более подробно процесс работы алгоритма. Обозначим каждую итерацию порядковым номером n ($n \in N$).

n = 1: Алгоритм начинает свою работу с авторизации пользователя по логину L и паролю P в клиентском программном обеспечении — торрент-клиенте (далее — клиент).

n = 2: Клиент вычисляет значение f_0 соотношения f_1 / f_2.

n = 3: Выполняем операцию подписи кортежа следующих параметров (Enc_K (L,P), f_1, f_2, f_0).

n = 4: В случае недействительности сертификата ЭП и/или закрытого ключа ЭП формируем и передаем запрос на новый сертификат ЭП в УЦ.

n = 4.1: УЦ принимает входящий запрос на сертификат ЭП.

n = 4.2: Выполняет проверку учетных данных.

n = 4.3: Реализует процедуру генерации нового сертификата ЭП, передает его клиенту (n = 3).

n = 5: Подписанный действующим сертификатом ЭП кортеж (Enc_K (L,P), f_1, f_2, f_0) передается трекеру.

n = 6: Трекер принимает переданный кортеж, выполняет проверку условия ($f_0 < 0$).

В случае, если условие из n = 6 не выполняется, переходим к n = 7, если выполняется — к n = 8.

n = 7: Трекер передает клиенту сообщение M2, что позволяет последнему свободно получать в кооперативное пользование искомые ресурсы.

n = 7.1: Клиент скачивает файл метаданных.

n = 7.2: Выполняется операция проверки действительности и корректности ЭП, которой подписан файл метаданных.

В случае, если ЭП недействительна, клиент возвращает ошибку и переходит к итерации n = 7.1.

В случае, если ЭП действительна, клиент начинает скачивать информацию в соответствии с содержимым файла метаданных. По завершении процесса алгоритм завершает свою работу.

n = 8: Поскольку условие ($f_0 < 0$) выполняется, клиент получает сообщение M1, что не позволяет ему сразу же начать процесс получения искомых ресурсов в кооперативное пользование.

В случае, если клиент не намерен продолжать работу по результатам итерации n = 8, он и алгоритм завершают свою работу.

n = 8.1: Клиент продолжает работу, ему необходимо передать в кооперативное пользование что-либо из имеющихся у него ресурсов.

n = 8.2: Клиент создает файл метаданных с информацией о тех ресурсах, которые он готов отдать в кооперативное пользование.

n = 8.3: Осуществляется операция подписания ЭП файла метаданных.

n = 8.4: Готовый файл метаданных публикуется на трекере.

Клиент переходит к итерации n = 2.

Заключение

На сегодняшний день процесс обмена данными между клиентом и сервером статистического учета (трекером) происходит недостаточно доверенным образом. Предложенный в работе алгоритм защиты учетных и статистических данных пользователей в распределенной вычислительной сети позволит существенно повысить уровень безопасности информации, передаваемой от клиента к серверу учета статистических данных (трекеру). Использование функционала удостоверяющего центра позволит дополнить существующий процесс обмена данными в распределенной вычислительной сети дополнительными компонентами, реализующими функции информационной безопасности.

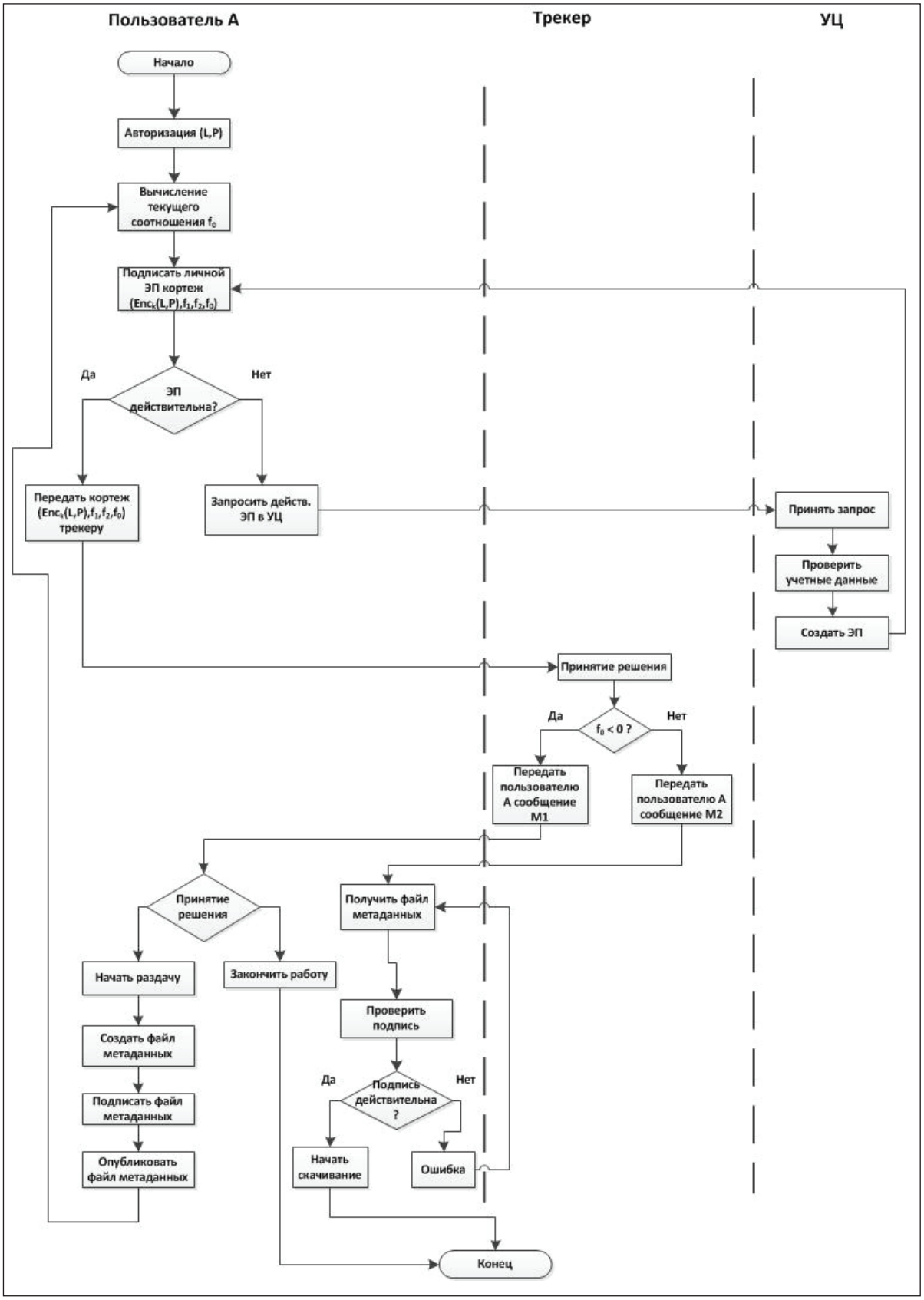

Рис. 1. Алгоритм защиты учетных и статистических данных

СПИСОК ЛИТЕРАТУРЫ:

1. BitTorrent. 2012 г. [Электронный ресурс]. URL: http://ru.wikipedia.org/wiki/BitTorrent.
2. The BitTorrent Protocol Specification. 2012 г. [Электронный ресурс]. URL: http://www.bittorrent.org/beps/bep_0003.html.
3. Torrent-файл. 2011 г. [Электронный ресурс]. URL: http://habrahabr.ru/post/119753/.