4. Facebook developers reference. https://developers.facebook.com/docs/reference/php/facebook-getSignedRequest
5. *Barth A., Jackson C., and Mitchell J.* Robust defences for cross-site request forgery // Proc. 15th ACM Conf. on Computer and Communications Security. ACM Press, 2008. P. 75–87.
6. ModSecurity Advanced Topic of the Week: HMAC Token Protection. http://blog.spiderlabs.com/2014/01/modsecurity-advanced-topic-of-the-week-hmac-token-protection.html
7. Understanding ASP.NET View State. http://msdn.microsoft.com/library/ms972976.aspx
8. NIST 800-162. Guide to Attribute Based Access Control (ABAC) Definition and Considerations. http://nvlpubs.nist.gov/nistpubs/specialpublications/NIST.sp.800-162.pdf

УДК 004.94

ОБ ИНФОРМАЦИОННЫХ ПОТОКАХ ПО ВРЕМЕНИ, ОСНОВАННЫХ НА ЗАГОЛОВКАХ КЭШИРОВАНИЯ ПРОТОКОЛА HTTP

Д. Н. Колегов, О. В. Брославский, Н. Е. Олексов

Рассматриваются информационные потоки по времени через заголовки кэширования протокола HTTP. Приводятся практические примеры реализации данных потоков и их основные характеристики, в частности достижимость на практике максимальной пропускной способности таких каналов при достаточно высоком уровне точности — 99,8 %.

Ключевые слова: *компьютерная безопасность, скрытые каналы, HTTP.*

Рассматривается задача анализа и реализации информационных потоков по времени, основанных на заголовках кэширования протокола HTTP. Обнаружение информационных потоков по времени и соответствующих им скрытых каналов в компьютерных системах является одной из задач компьютерной безопасности [1]. Распространённость протокола HTTP делает выявление методов реализации информационных потоков по времени перспективным направлением для исследований.

Известные на данный момент информационные потоки в протоколе HTTP, как правило, основываются на отправке HTTP-запросов со специальными GET- или POST-параметрами или на применении стеганографических методов для сокрытия факта передачи информации в HTTP-заголовках. Однако данные методы меняют стандартную структуру HTTP-запроса, а значит, требуют соответствующей модификации веб-сервера. Информационные потоки, рассматриваемые в данной работе, не накладывают дополнительных ограничений на конфигурацию веб-сервера и, следовательно, представляют больший интерес для изучения.

Заголовки кэширования в протоколе HTTP хранят информацию о времени последнего изменения веб-страницы, тем самым позволяя клиенту не загружать веб-страницу, если она не была изменена с момента последнего запроса. Таким образом, возможна передача информации на основании данных об изменениях запрашиваемой веб-страницы. Например, ситуация, при которой некоторая веб-страница была изменена с момента последнего обращения к ней, может быть интерпретирована как получение одного бита информации.

Рассмотрим общую схему информационного потока по времени данного типа (рис. 1). Пусть O_1 — объект, доступный на чтение процессу S_1 на $host_1$; O_2 — веб-ресурс, расположенный на $host_1$; O_3 — HTTP-ответ; O_4 — объект, доступный на запись

процессу S_3 на $host_2$; S_3 — веб-сервер; S_1 и S_2 — два процесса, которым в соответствии с политикой безопасности данной компьютерной системы запрещено общаться напрямую. В каждый момент времени процесс S_1 считывает один бит информации из объекта O_1 и, в зависимости от значения бита, осуществляет или не осуществляет доступ на запись к веб-странице O_2. Процесс S_3 выполняет HTTP-запрос к O_2 и после получения HTTP-ответа O_3, основываясь на изменениях сущности O_2, пишет в объект O_4 соответствующий бит (например, 1, если веб-страница была изменена, и 0 в противном случае).

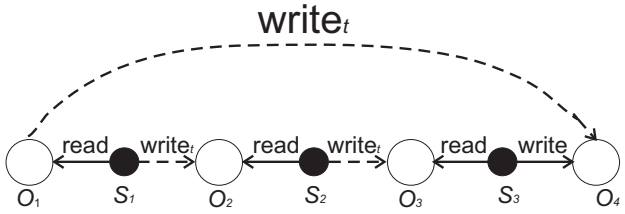

Рис. 1. Общая схема информационного потока

Информационные потоки по времени, основанные на заголовках кэширования протокола HTTP, могут быть разделены на две группы: потоки, основанные на заголовке Last-Modified, и потоки, основанные на заголовке ETag.

Заголовок Last-Modified содержит время последнего изменения сущности на веб-сервере, например, «Last-Modified: Tue, 10 Apr 2014, 12:34:56 GMT». Существует три возможных варианта реализации потока по времени на основе заголовка Last-Modified:

1) по значению заголовка Last-Modified: S_3 обращается к O_2 и получает HTTP-ответ O_3; сравнивая полученное в O_3 значение заголовка со значением, полученным в предыдущий раз, S_3 делает вывод об отправленном бите — если значения не совпадают, то получена 1, иначе 0;
2) с помощью заголовка If-Modified-Since: S_3 обращается к O_2 с заголовком «If-Modified-Since: $Date_1$» и получает HTTP-ответ O_3. Если код ответа O_3 равен 200, значит, веб-страница O_2 была изменена и получена 1; если код ответа 304, то веб-страница не менялась и получен 0;
3) с помощью заголовка If-Unmodified-Since: S_3 обращается к O_2 с заголовком «If-Unmodified-Since: $Date_1$» и получает HTTP-ответ O_3. Если код ответа O_3 равен 412, значит, веб-страница O_2 была изменена и получена 1; если код ответа 200, то веб-страница не менялась и получен 0.

Для реализации рассматриваемых информационных потоков по времени необходимо, чтобы S_1 имел права на запись в объект O_2 и на чтение из объекта O_1; S_3 имел права на чтение O_2, то есть мог обратиться к сущности O_2 на веб-сервере через HTTP. При стандартной конфигурации веб-сервера информационные потоки, основанные на заголовке Last-Modified, имеют одинаковую теоретическую максимальную скорость 1 бит/с. Данная скорость обусловлена техническими ограничениями заголовка Last-Modified: формат даты, используемый заголовком, хранит время с точностью до секунд.

В результате тестирования реализации одного из вышеописанных потоков установлено, что максимальная скорость потока достижима на практике, если пропускная способность канала между S_1 и S_3 позволяет S_3 сделать запрос к O_2 и получить ответ O_3 за 1 с. Точность передачи для полученной реализации составила 99,82 %.

Заголовок ETag (entity tag) используется для контроля изменений HTTP-сущностей. Среди наиболее распространенных веб-серверов только в Apache стандартизован алгоритм формирования заголовка ETag [3]. Значение ETag в соответствии с данным алгоритмом формируется из шестнадцатеричных значений идентификатора сущности (inode), размера сущности и времени последнего изменения сущности в формате mtime: «ETag: 120c7bL-32bL-4f86d4105ac62L». Соответственно могут быть рассмотрены три варианта информационных потоков, основанных на заголовке ETag:

1) по значению заголовка ETag: S_3 обращается к O_2, получает HTTP-ответ O_3 и делает вывод об отправленном бите, сравнивая полученное в O_3 значение заголовка со значением, полученным в предыдущий раз;
2) с помощью заголовка If-Match: S_3 обращается к O_2 с заголовком «If-Match: $ETag_1$» и получает HTTP-ответ O_3. Если код ответа O_3 равен 412, значит, получена 1; если код ответа 200, то получен 0;
3) с помощью заголовка If-None-Match: S_3 обращается к O_2 с заголовком «If-None-Match: $ETag_1$» и получает HTTP ответ O_3. Если код ответа O_3 равен 200, значит, получена 1; если код ответа 304, то получен 0.

Реализация данных потоков требует прав доступа, аналогичных требованиям потока по заголовку Last-Modified. В стандартной конфигурации веб-сервер обновляет значения ETag не чаще чем раз в секунду, то есть, если клиент запрашивает веб-страницу более одного раза в секунду, все ответы будут содержать одно и то же значение заголовка ETag. Таким образом, при сохранении исходной конфигурации максимальная скорость потока не может превышать 1 бит/с, как и в случае с потоком по Last-Modified. Однако, в отличие от Last-Modified, ETag хранит время в формате mtime, точность которого составляет 1 мкс. Следовательно, теоретическая пропускная способность потока равна приблизительно 976 кбит/с, что намного превышает пропускную способность, достижимую на практике.

Чтобы максимально эффективно использовать пропускную способность данных потоков, необходимо, чтобы значение заголовка обновлялось при каждом изменении сущности. Для этого возможно изменить конфигурацию веб-сервера S_2 или изменить тип сущности O_2 со статической (html) страницы на динамическую (php). PHP позволяет вручную задавать значения и вид заголовков HTTP-ответа для отдельной страницы; соответственно возможно генерировать неотличимое от настоящего значение ETag (используя тот же алгоритм формирования), но делать это для каждого HTTP-запроса. Таким образом, максимальная скорость для данных потоков равна 1 бит в $(2L + T)$ секунд, где L — время, затрачиваемое на передачу сообщения между S_2 и S_3, и T — время, необходимое S_1 и S_3 для вычислительных операций: сравнения значений заголовков, чтения и записи битов и т. п. Реализации данных потоков при тестировании на скорости 2 бит/с показали 99,56 % точности передачи.

ЛИТЕРАТУРА

1. CWE-385: Covert Timing Channel. `https://cwe.mitre.org/data/definitions/385.html`
2. *Brown E., Yuan B., Johnson D., and Lutz P.* Covert channels in the HTTP Network Protocol: Channel characterization and detecting Man-in-the-Middle Attacks // Proc. 5th Intern. Conf. Information Warfare and Security. Ohio, USA, April 8–9, 2010. Air Force Institute of Technology, 2010. P. 56–65.
3. ETag header Apache specification. `http://httpd.apache.org/docs/2.2/mod/core.html#fileetag`