pCloud.com

Възможности, които всички web услуги трябва да имат

. . . а са рядко срещани

Васил Колев <vasil@pcloud.com>

pCloud.com

Идеята за тази лекция

- Огромно количество проблеми, които се случват
- Лесни начини за решаване
- Хората отказват да погледнат историята
- Нищо особено ново под слънцето
  - (моля не заспивайте)

pCloud.com

Защо ние го говорим това

- Опитваме се да не сме от многото, които повтарят предишни грешки

pCloud.com

Кои сме ние

- pCloud - cloud storage, направен правилно
- удобен, ефективен, бърз
- https://pcloud.com/ , https://docs.pcloud.com/
- https://github.com/pcloudfs

Authentication

Authentication & Password storage

- Най-простото нещо - влизане в системата

Стандартен метод за auth

- Пращате username и парола
- Получавате отговор/token
- Подслушването е тривиално

Правилен метод

- Сървърът ви дава nonce/challenge/както-там-се-казва
  - random нещо, което не се повтаря повече
- Изчислявате HMAC_SHA1(password.nonce)
- Пращате това на сървъра
- Сървъра изчислява същото сравнява и знае дали сте вие
- Тривиално за правене, малко по-бавно
  - Не е нещо, което трябва да е особено бързо
- Дава възможност и клиентът да разпознава сървъра

pCloud.com

Съхраняване на пароли

Съхраняване на пароли

трябва да е ясно и лесно...

...ама друг път

Plaintext в базата данни

- Пазенето на парола в plaintext е глупаво
- Все още много услуги го правят
- Hack-ват по няколко годишно
  - преди седмица-две - ebay

Първи лесен (и грешен) вариант

- MD5(password)
- или за по-сигурно SHA1(password)
- . . . като и двете подлежат на атаката с rainbow таблиците

Втори лесен (и правилен) вариант

- crypt() със salt
- представлява salt.HMAC_SHA1(password.salt)
- не се поддава на атака с rainbow таблица
- не се поддава на чупене по лесен начин
- не може да се познае дали паролата ви за един service е същата като в друг
- . . . но не може да се направи challenge-response auth.

pCloud.com

Трети не-толкова-лесен (но правилен) вариант

- Частичен hash
  - т.е. запазване на state на hash функцията
- Решава проблема с rainbow таблиците
- Дава възможност за challenge-response
- Отнема малко повече време да се реализира

Четвърти (велик и труден за правене) вариант

- Измислен от Дан Каминski
  - password-authenticated key agreement
- На база на паролата се генерират двойка public/private key
- Сървърът съхранява само public key
- Клиентът винаги може да повтори действието с паролата
- Ако ви изтече базата, даже няма да могат да се логнат при вас с информацията
- Няма публична реализация
- http://dankaminsky.com/2012/01/03/phidelius/

Further reading

- PBKDF2 https://en.wikipedia.org/wiki/Pbkdf2
- https://password-hashing.net/

Комуникация

Как най-добре да си говорим с услугите

- . . . или защо web-а е тъпа брадва

Стандартни неща

- Всичко ползва JSON/HTTP
- Бавни за parse-ване, текстови протоколи
- Много overhead
- Липса на pipelining, контрол

Трябва алтернатива

- Не всичките ви клиенти са browser-и
  - т.е. глупави приложения, които трябва да поддържат огромно количество глупости
- Дайте възможност на хората да ползват нещо ефективно
- Вече има webrtc и websockets

Примерен binary протокол

- pCloud има собствен binary протокол
  - https://docs.pcloud.com/binary_protocol/index.html
- Елементарен за работа и бърз
- 700 реда C, 536 реда на java (с коментарите)
- Има алтернативи, като google protobuf, msgpack и т.н.

QUIC и пътят напред

- Протокол на google, излязъл в средата на 2013
- Замества TCP и TLS, т.е. има вградено криптиране
- Осъществяването на връзка става за 1 RTT
  - TCP+TLS отнема 3-4 rtt
- Multiplexing, forward error correction
- Ако се ползва достатъчно, няма да е проблем с firewall-ите.
- Около 88к реда C++ код с тестовете
  - TCP - 22k SSL - 55k
  - Нищо не може да е толкова омазано, колкото SSL/TLS
- https://en.wikipedia.org/wiki/QUIC

Употреба на SSL/TLS

- Edward Snowden
- Вече все повече услуги предлагат SSL/TLS
  - Не всички
- Не всички съвсем правилно
- (SSL е стария протокол, правилният е TLS1.2)

Правилна употреба на SSL/TLS

- Шифри/протоколни версии
  - Не бъдете като java/android
  - http://op-co.de/blog/posts/android_ssl_downgrade/
- Forward secrecy
- https://github.com/ioerror/duraconf за конфигурации
- https://www.ssllabs.com/ssltest/ за тестване

SSL/TLS Session caching

- Спестява rtt на повторни връзки
- Кеширане на параметри/сертификати
- http://vincent.bernat.im/en/blog/2011-ssl-session-reuse-rfc5077.html
- По RFC 5246, не по RFC 5077 (при което се губи forward secrecy)

Privacy

- Разкриване дали някой ни е потребител
- Разкриване дали имаме даден файл

Не всички клиенти са равни

- Това, което е удобно в web, не е удобно в телефоните
- Дори това, което е удобно в Chrome, не е удобно в IE
- Не съществува ЕДИН ПРАВИЛЕН НАЧИН
  - колкото и да го твърдят някои хора

Предаване на параметри

- Защо не навсякъде?
- GET, POST, Cookie
- Дава възможност за прострелване в крака
  - не искате auth в get и да refer-нете навън

обработка на request-и в движение

- Очевидна латентност - чака се цялата заявка, преди да се обработи
- Повечето content може да се process-ва, докато се upload-ва

blatant self-promotion

Специфични неща за cloud услугите

- Оставащата част си е живо фукане от наша страна

Thumbnail combining

- Много малки файлчета в един отговор
- За HTTP - грозно, но работи

On-the-fly zip

- Архивите се създават в движение
- Няма изчакване да се генерира

Rsync-подобен протокол

- Спестява bandwidth и време
- В двете посоки

Видео с напасващ се bit rate

- С колкото може да пише към вас, с толкова encode-ва

Файлови операции

- Възможност за бърникане директно по файловете
- Интерфейс като файлова система

Други готини неща за в бъдеще

- end-to-end crypto
- ...

pCloud.com

Отворени сме към други идеи

- Винаги сме отворени към подобряване на нашата услуга
- Приемаме и критика :)

pCloud.com

Благодаря!

- Благодаря за отделеното време!
- Въпроси?