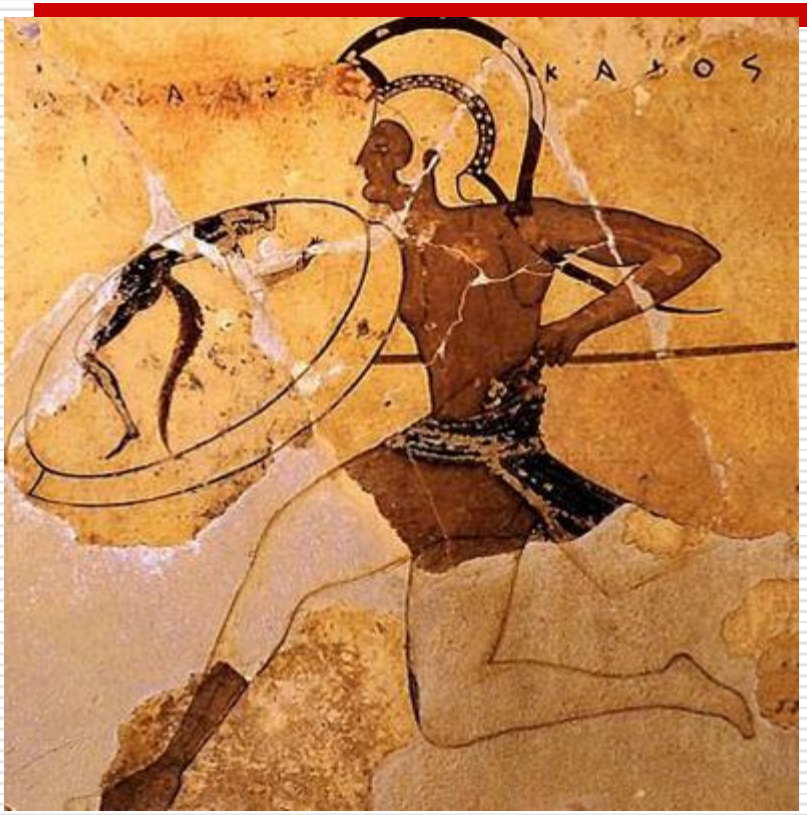
Современные криптографические протоколы

Криптографически защищенные коммуникации

Немного истории



- Общение между пользователями интернета:
  - BBS – конец 1970-х
  - SMTP – 1982
  - IRC – 1988
  - ICQ – 1996
- Криптография:
  - PGP - 1991

Вернёмся к вопросам защиты коммуникаций

- Нешифрованные – кто угодно с доступом к сети может читать содержимое
- Неаутентифицированные – кто угодно может выдать себя за кого угодно
- Но… **есть отказуемость**

PGP

Ставим GnuPG, генерим себе ключевую пару - «публичный ключ» и «приватный ключ».

Приватный ключ используется по аналогии с блочным закрытым – ЭЦП и дешифровка входящих сообщений

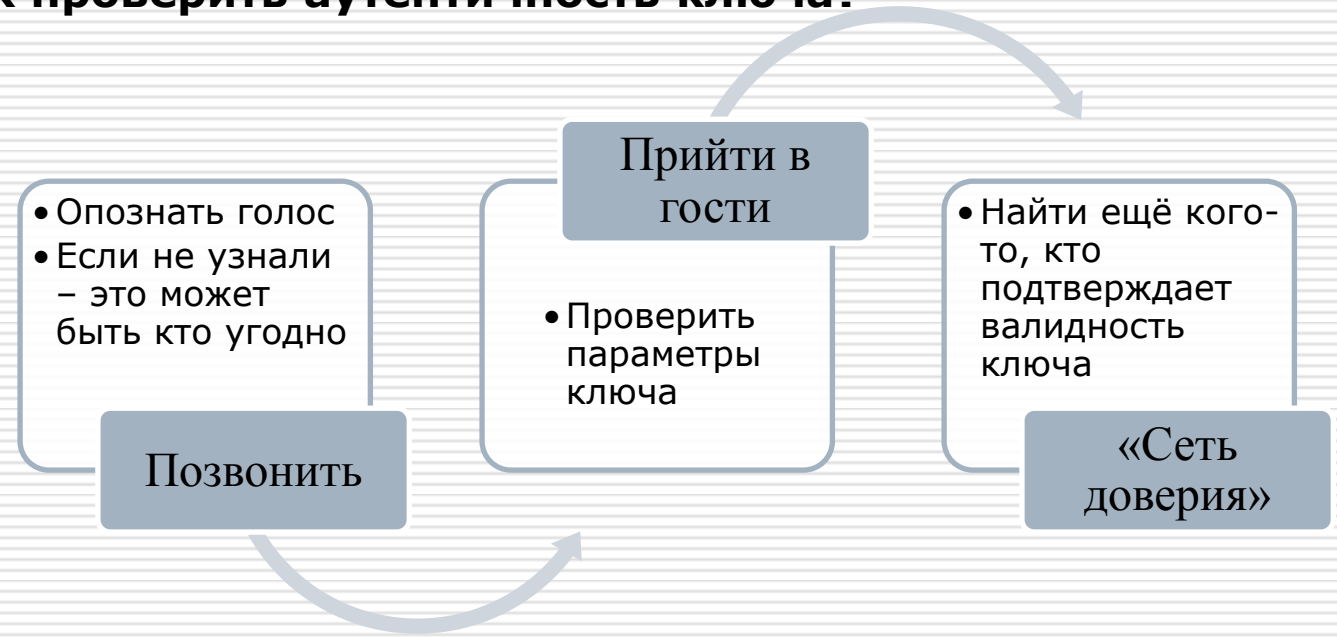
Публикуем «публичный» PGP ключ, отправляя его на сервер ключе в интернете.

Люди используют наш публичный ключ, чтобы зашифровать отправленные нам сообщения.

Приватный ключ есть только у нас, так что только мы можем прочитать (или подписать) сообщения.

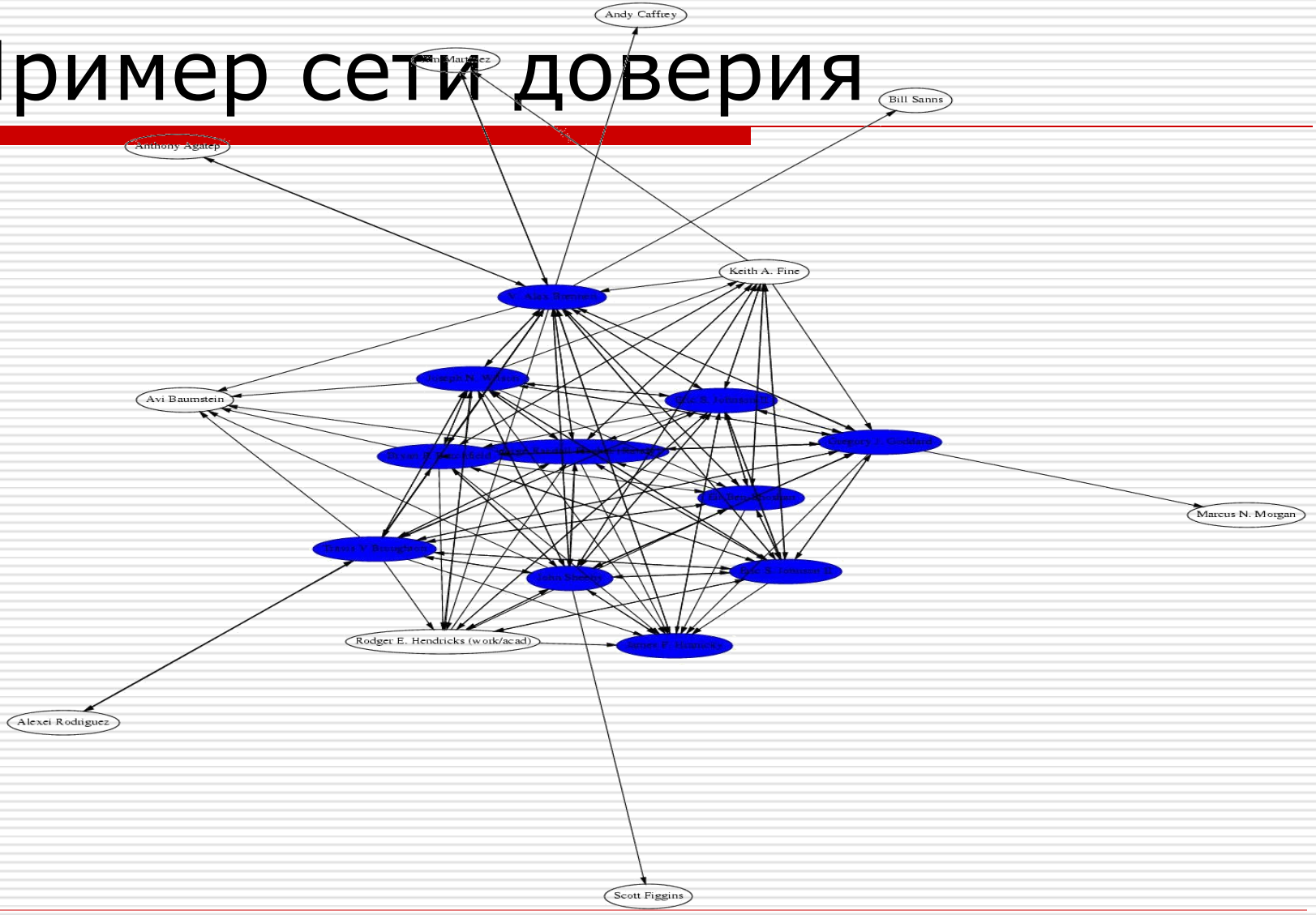
Сеть доверия PGP

- Кто угодно может загрузить ключ на сервер ключей – в том числе **поддельный ключ**
- **Как проверить аутентичность ключа?**



- После проверки принадлежности ключа его можно подписать, публично подтверждая, что вы выполнили проверку.

Пример сети доверия



“Trolling WoT”

Вот перед вами ключ. Вы бы его подписали?

```
pub   1024D/1B629B3D 2005-12-27
      Key fingerprint = 965E F829 EA6C 9174
   4B46  43E1 4513 9A86 1B62 9B3D
uid                  u1tr4 l4s3r
   <seekrit@hax0r.com>
sub   2048g/1F8E2EEA 2005-12-27
```

Что вам нужно узнать, прежде чем вы подпишете?

“Trolling WoT”

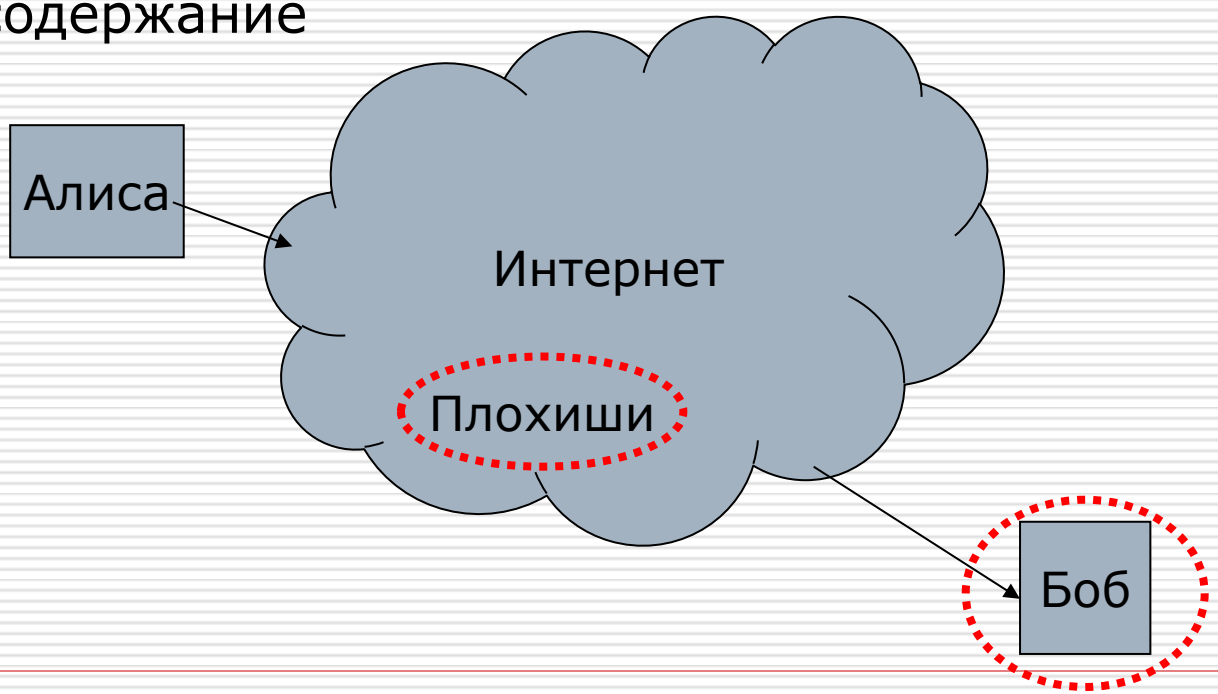
- Доклад на конференции OHM2013 о поддельных PGP ключах
  - https://www.eff.org/event/ohm2013-trolling-web-trust
- Утилита для генерации: https://github.com/micahflee/trollwot
  - Добавление «поддельных» подписей к ключам
  - Подбор PGP key id (и отпечатка)
  - Создание поддельных ключей по набору имён и адресов электронной почты, и генерация сети доверия из них

Проблемы с PGP

- Недружелюбный формат отпечатка
  - Сложно запоминать и произносить вслух
    - Pseudo-word fingerprints https://github.com/trevp/keyname
- Поддельные сети доверия
- Отсутствие секретности передачи
  - Украденные ключи компрометируют всю предыдущую переписку

Модель для защищенных коммуникаций

- Предположения
  - Ализа и Боб оба умеют пользоваться PGP
  - Они знают ключи друг друга
  - Они не хотят спрятать факт коммуникаций, только содержание



А теперь «плохиши»

- Компьютер Боба украден
  - Преступники, конкуренты
  - Удержан в качестве вещественного доказательства
- Или незаметно «логически» захвачен
  - Вирус, троян, кейлоггер
- *Весь* ключевой материал утёк
  - О нет!
- Плохиши теперь могут:
  - Расшифровать всю старую переписку
  - Изучить её содержимое
  - Узнать, что автором была Алиса
    - И у них есть *математическое* доказательство для всего этого
- Насколько такое общение можно назвать приватным?

На базе PGP сделано много вариантов защищенных коммуникаций

- Электронная почта с PGP
- Клиенты систем мгновенного обмена сообщениями:
  - Jabber (Pidgin и т.д.)
  - ICQ/AIM
  - Вообще говоря, любая система IM может быть транспортом для PGP-MIME
- Есть даже реализации WoT для WWW и OpenSSH
  - http://web.monkeysphere.info/

SILC

- Аббревиатура от **S**ecure **I**nternet **L**ive **C**onferencing.
- Создавался как замена IRC (**I**nternet **R**elay **C**hat), выпущен в 2000 г.
- Есть функциональность IM.
- Доступны реализации сервера и клиента (http://www.silcnet.org)

Протокол SILC

- **Сервер** поддерживает каналы и обрабатывает соединения от клиентов
- **Клиент** приходит на сервер, чтобы подключаться и отключаться от каналов
- **Канал** это группа клиентов, которые разделяют один и тот же разговор
- Никто за пределами канала не должен иметь возможность прослушивать разговор
- Предполагается, что каждый клиент согласовал сессионный ключ со своим сервером

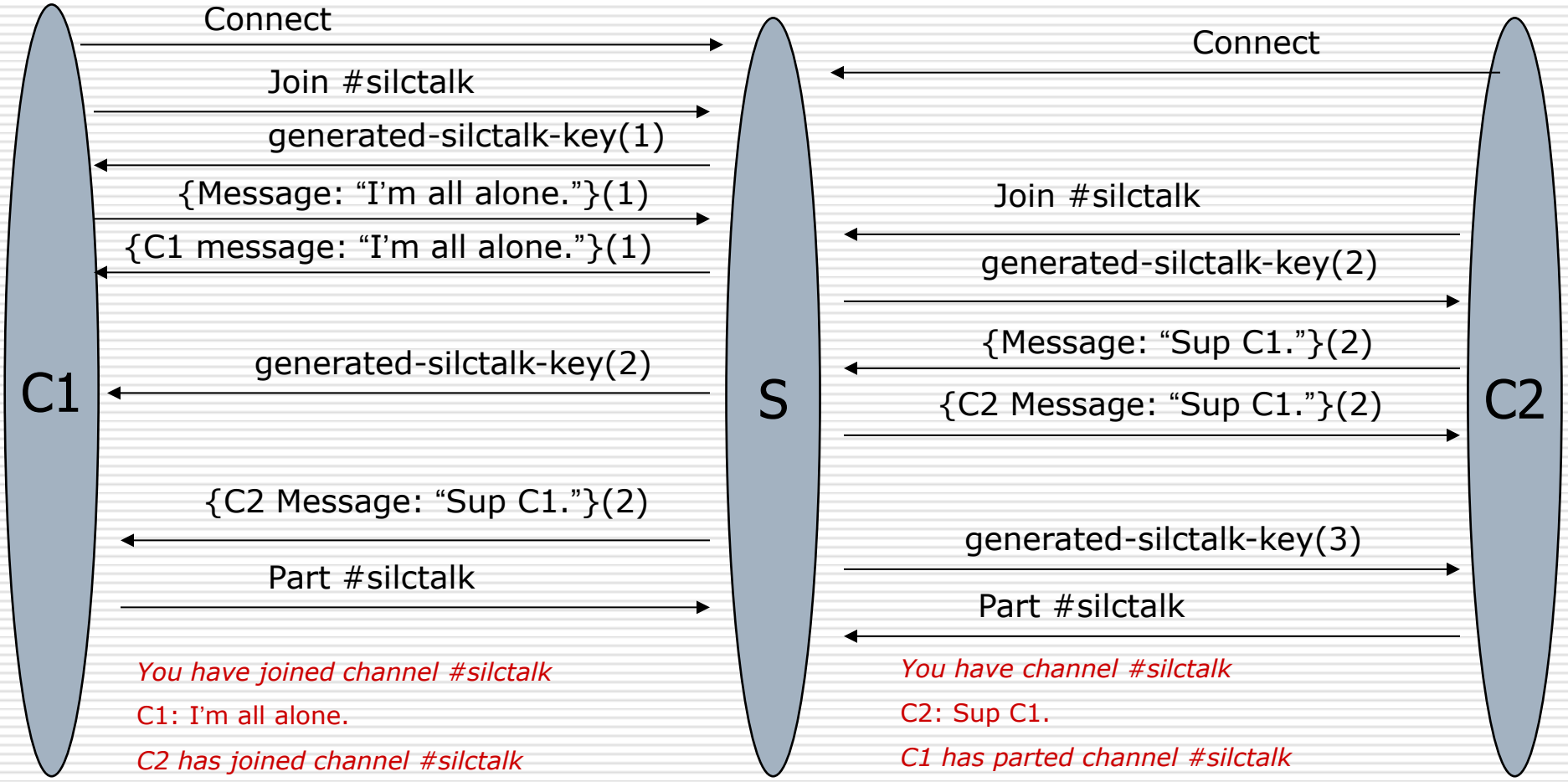
Клиентская часть протокола

- Если сущность A отправляет что-то сущности B через SILC, оно всегда шифруется сессионным ключом на пути от A к B
- Клиент инициирует соединение к серверу
- Соединившись, клиент может попросить у сервера доступ к каналу
- Клиент знает, что подключился к каналу, когда получил от сервера ключ канала
- Каждый раз при входе и выходе клиента генерируется новый ключ канала, который рассылается оставшимся клиентам
- Каждое сообщение на канале шифруется ключом канала, а не сессионным ключом. В то же время, заголовки шифруются ключом сессии.
- Клиент при уходе с канала уведомляет сервер, что он может обновить список канаал и перегенерировать ключ.

Серверная часть протокола

- При получении запроса на доступ к каналу от клиента, добавляет его в список канала
- При получении запроса на выход с канала, удаляет клиента из списка канала
- При изменении списка заново генерируется ключ канала
- При получении сообщения от клиента на канале, оно пересылается всем остальным (только заголовки шифруются заново для каждого клиента)

Пример


C1
S
C2
Connect
Join #silctalk
generated-silctalk-key(1)
{Message: “I’m all alone.”}(1)
{C1 message: “I’m all alone.”}(1)
generated-silctalk-key(2)
{C2 Message: “Sup C1.”}(2)
Part #silctalk
Connect
Join #silctalk
generated-silctalk-key(2)
{Message: “Sup C1.”}(2)
{C2 Message: “Sup C1.”}(2)
generated-silctalk-key(3)
Part #silctalk
You have joined channel #silctalk
C1: I’m all alone.
C2 has joined channel #silctalk
C2: Sup C1.
You have parted channel #silctalk
You have channel #silctalk
C2: Sup C1.
C1 has parted channel #silctalk
You have channel #silctalk

Секретность пересылки (forward secrecy)

- SILC перегенерирует ключ канала при каждом заходе/уходе
- Пользователи могут дополнительно согласовать постоянный ключ канала
  - Сообщения канала не видны серверу
  - Ключами управлять сложно

OTR

- Предложен Йэном Голбергом и Никитой Борисовым в 2004 г.
- Ключевые свойства:
  - Абсолютная секретность передачи
  - Отказуемость

Модель для OTR: приватное общение в реальной жизни

- Алиса и Боб общаются в закрытом помещении
- Никто больше не слышит
  - Если не ведётся запись
- Никто не знает содержания беседы
  - Если Алиса или Боб не скажут
- Никто не может *доказать,* что было сказано
  - Даже Алиса или Боб

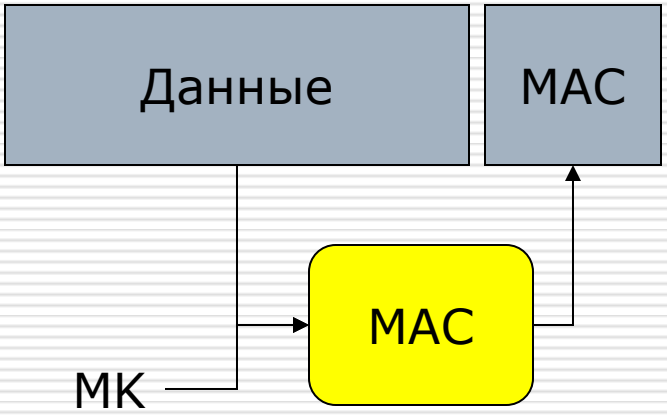
Абсолютная секретность передачи

- Короткоживущий ключ шифрования
- Шифрование этим ключом
- Выбросить после использования
  - Надёжно удалить из памяти
- Долгоживущие ключи для аутентификации и создания шифрованного канала для выработки временного ключа

Отказуемая аутентификация

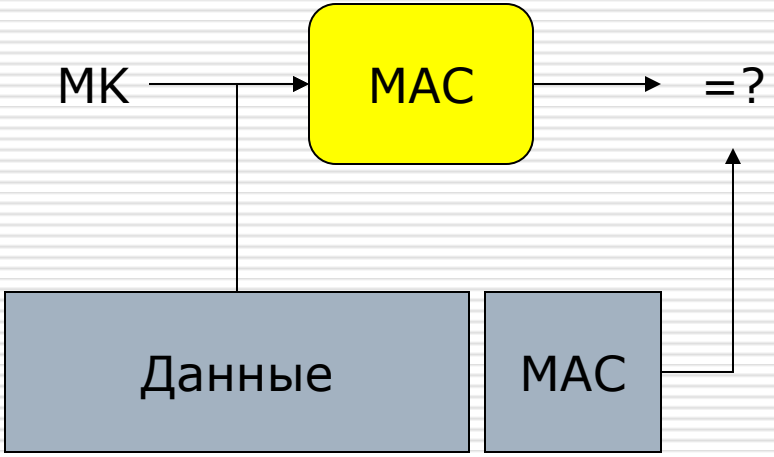
- *Нет* ЭЦП
  - Отставим неотказуемость для контрактов, а не для разговора
- *Есть* аутентификация
  - Нельзя обеспечить секретность, если кто угодно может выдать себя за вашего приятеля
- Используем Message Authentication Codes (MAC)

MAC в OTR



Алиса

Боб



Нет доказательной базы для третьей стороны

- Аутентификация на разделяемом секрете
  - Алиса и Боб разделяют общий МК
  - МК нужен для вычисления MAC
- Боб не может доказать, что Алиса сгенерировала MAC
  - Он тоже мог это сделать
  - Каждый, кто может проверить, может и подделать

Протокол OTR фаза 1: AKE

- Алиса и Боб выбирают случайные x, y resp.
- A->B: g^x, $Sign_{Alice}(g^x)$
- B->A: g^y, $Sign_{Bob}(g^y)$
- SS=g^{xy} - разделяемый секрет
- Подписи аутентифицируют разделяемый секрет, а не сообщения

OTR фаза 2: передача сообщения

- Вычислить EK=Hash(SS), MK=Hash(EK)
- A->B: $Enc_{EK}(M)$, $MAC(Enc_{EK}(M),MK)$
- *Enc* - блочный шифр (AES)
- Боб проверяет MAC используя MK, расшифровывает M используя EK
- Обеспечены конфиденциальность и аутентичность

OTR: перегенерация ключей

- Алиса и Боб выбирают x′,y′
- A->B: $g^{x'}$, MAC($g^{x'}$, MK)
- B->A: $g^{y'}$, MAC($g^{y'}$, MK)
- SS′ = H($g^{x'y'}$)
- EK′ = H(SS′), MK′=H(EK′)
- Алиса и Боб надёжно удаляют SS, x, y и EK
  - **Абсолютная секретность передачи**

Ограничения OTR

- Существенно интерактивный
  - Короткое время жизни ключа
  - Создан для IM
- Фундаментально ограничен двумя участниками
  - Отказуемый множественный OTR – сложная задача

mpOTR

- Multy-party Off-the-record communications
- Идеи предложены Йэном Голдбергом в 2009
- Текущее состояние:
  - https://moderncrypto.org/mailman/listinfo/messaging
  - http://lists.cypherpunks.ca/mailman/listinfo/otr-dev
  - http://mpotr.secsem.ru/
- Первая реализация ожидается в 2014 г.

- Установление канала
  - IRC, XMPP MUC
- Аутентификация и выработка общего секрета
  - Групповой DH
- Общение
  - Проблема сохранения порядка сообщений и причинности
- Разрыв канала
  - Публикация эфемерных ключей

Другие существующие проекты

- TorChat
  - Основан на скрытых сервисах TOR
- CryptoCat
  - https://blog.crypto.cat/wp-content/uploads/2012/11/Cryptocat-2-Pentest-Report.pdf
  - Сейчас внутри использует OTR, распространяется как браузерный плагин
  - Активно разрабатывают mpOTR (mpCat)
- Gibberbot, TextSecure, Xabber – Android
- ChatSecure - iOS

Задание

- Установить OTR плагин для Pidgin
- Согласовать с коллегой оффлайновый «секрет»
- Надёжно идентифицировать друг друга в OTR с помощью этого секрета
- Изучить протокол SMP (Socialist Millionaire)