Fundamentos de Segurança Informática (FSI)

2021/2022 - LEIC

Manuel Barbosa
mbb@fc.up.pt

Aula 23
TLS e Signal

Transport Layer Security (TLS)

Modelo de Segurança

- Atacante na rede (já vimos):
  - controla a infra-estrutura: routers, DNS, etc.
  - Escuta, injeta, bloqueia, altera pacotes/mensagens.
- Exemplos:
  - rede sem fios num café/loja
  - acesso a rede num hotel/empresa
  - o nosso ISP …

Handshake TLS1.3 = Diffie-Hellman Autenticado

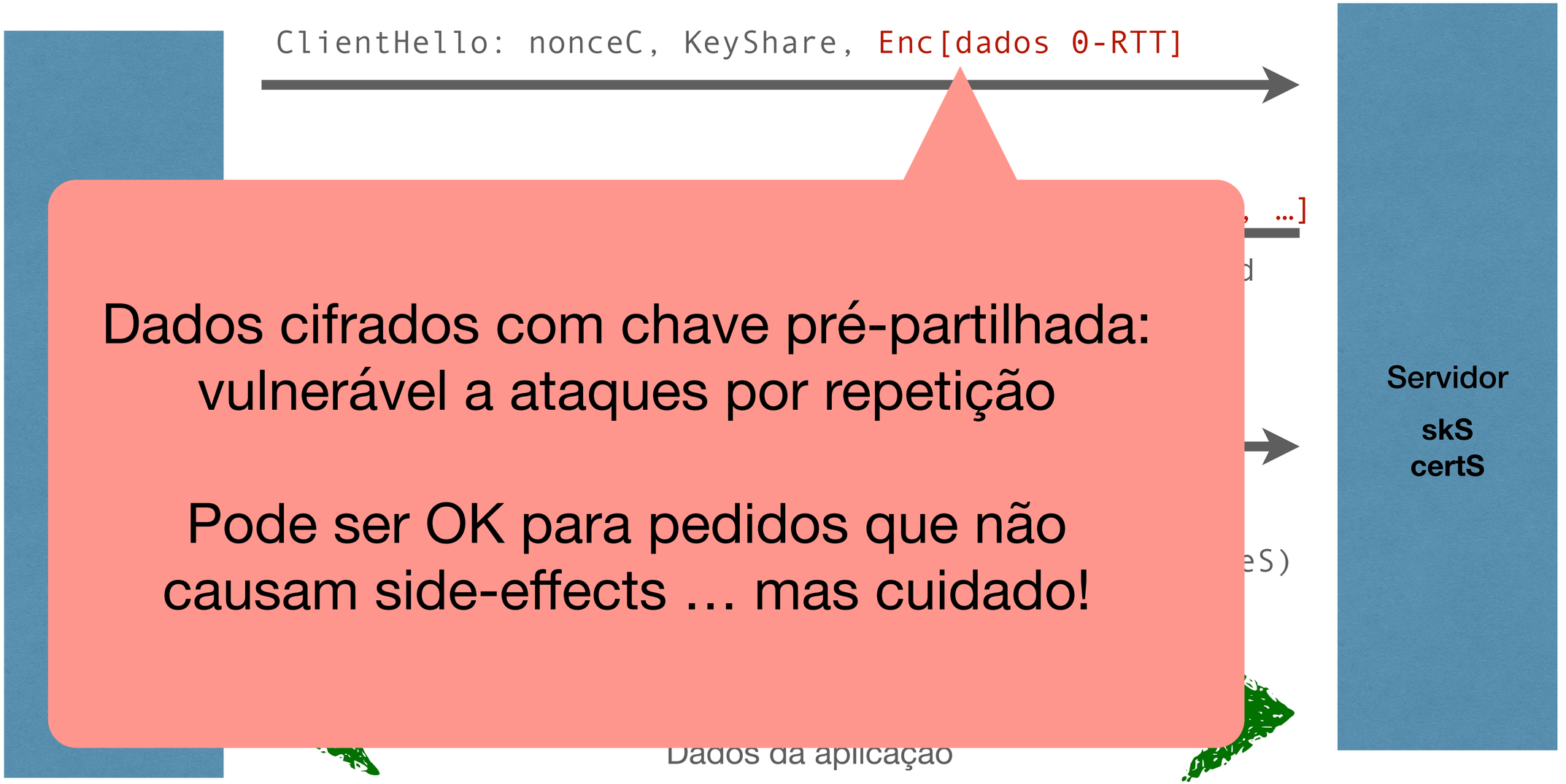
- Historicamente o handshake TLS utilizava maioritariamente:
  - transporte de uma chave de sessão do cliente para o servidor (RSA)
  - autenticação implícita do servidor => utilização correta da chave de sessão
- Hoje em dia tornaram-se evidentes as vantagens DH:
  - eficiência com utilização de curvas elípticas
  - perfect forward secrecy:
    - chaves de longa duração são utilizadas para assinaturas e não para transporte de chaves de sessão
    - comprometer uma chave de longa duração não compromete acordos de chave passados

Utilização do TLS implica PKI

- O servidor autentica a troca DH com uma assinatura digital (assinatura cliente = opcional)
- Como é que o cliente conhece a chave de verificação do servidor?
  - Servidor envia o seu certificado de chave pública
  - Cliente verifica validade do certificado (root CAs pré instaladas) =>
    - nome de domínio faz match com subject do certificado
    - é possível utilizar um wildcard no item mais à esquerda do DN: e.g., *.a.com
  - Cliente utiliza chave pública no certificado para verificar assinatura digital no protocolo DH
- Recordar: cliente não autenticado => servidor não sabe com quem fala/estabeleceu chave

Handshake TLS 1.3

Handshake TLS 1.3: Optimização (cautela)



Dados cifrados com chave pré-partilhada:
vulnerável a ataques por repetição

Pode ser OK para pedidos que não
causam side-effects … mas cuidado!

Integração TLS/HTTP

- As mensagens HTTP são transmitidas usualmente como payloads TLS depois do canal estabelecido.
- Problemas/soluções:
  - web proxy: proxy precisa de saber o cabeçalho HTTP para estabelecer ligação
    - cliente envia nome de domínio antes do client-hello do TLS
  - virtual hosts: mesmo IP com múltiplos DNS => como sabe o servidor que certificado devolver?
    - solução antiga: client-hello inclui nome de domínio do servidor
    - TLS1.3 tenta preservar privacidade do nome de domínio (certificado cifrado)
      - solução futura: nome de domínio cifrado com chave pública proveniente do DNS

Integração TLS/HTTP

- Porque não utilizar sempre HTTPS para todo o tráfego?
  - antigamente => performance
  - hoje em dia => AES-NI => aceleradores de HW => não há desculpas
- Desde 2018 browsers tendem a catalogar sites HTTP como inseguros
  - e.g., sinais visuais, alarmes quando há envio de credenciais, etc.

TLS/HTTPS nos browsers

- Duvidoso se os indicadores visuais são suficientes:
  - confirmamos o nome de domínio?
  - confirmamos que o aloquete está ativo?
- E se nos convencem a fazer uma ligação http://www.paypal.com?
  - política correta do servidor => redirecionar pedidos http para https
  - mas => um MiM poderia ligar-se em nosso nome a https://www.paypal.com
  - isto chama-se um SSL Strip Attack (browsers modernos apresentam avisos => utilizador?)
  - solução => possível incluir nos cabeçalhos informação de que todas as ligações futuras devem ser feitas por HTTPS (Strict Transport Security) => desaparece com limpeza da cache

TLS/HTTPS nos browsers

- Nunca esquecer os ataques Man in the Middle:
  - só são impedidos se conseguirmos autenticar chave pública do servidor
- O impacto de uma Autoridade de Certificação corrompida é devastador:
  - vários exemplos TurkTrust (2013), Indian NIC (2014), WoSign (2016)
  - assinaram certificados para Google, Yahoo, etc.
- Só se pode corrigir eliminando certificados dos sistemas operativos/ browsers

TLS/HTTPS nos browsers

- Potenciais problemas de mixed content HTTP/HTTPS
- Nunca utilizar objetos embebidos a partir de links HTTP:
  - Exemplo: ir buscar um script JavaScript a http://muitoutil.com
  - Atacante pode substituir o script!
- Browsers hoje em dia também avisam sobre mixed content
  - E.g. já vimos submissão de forms sem HTTPS

Privacidade

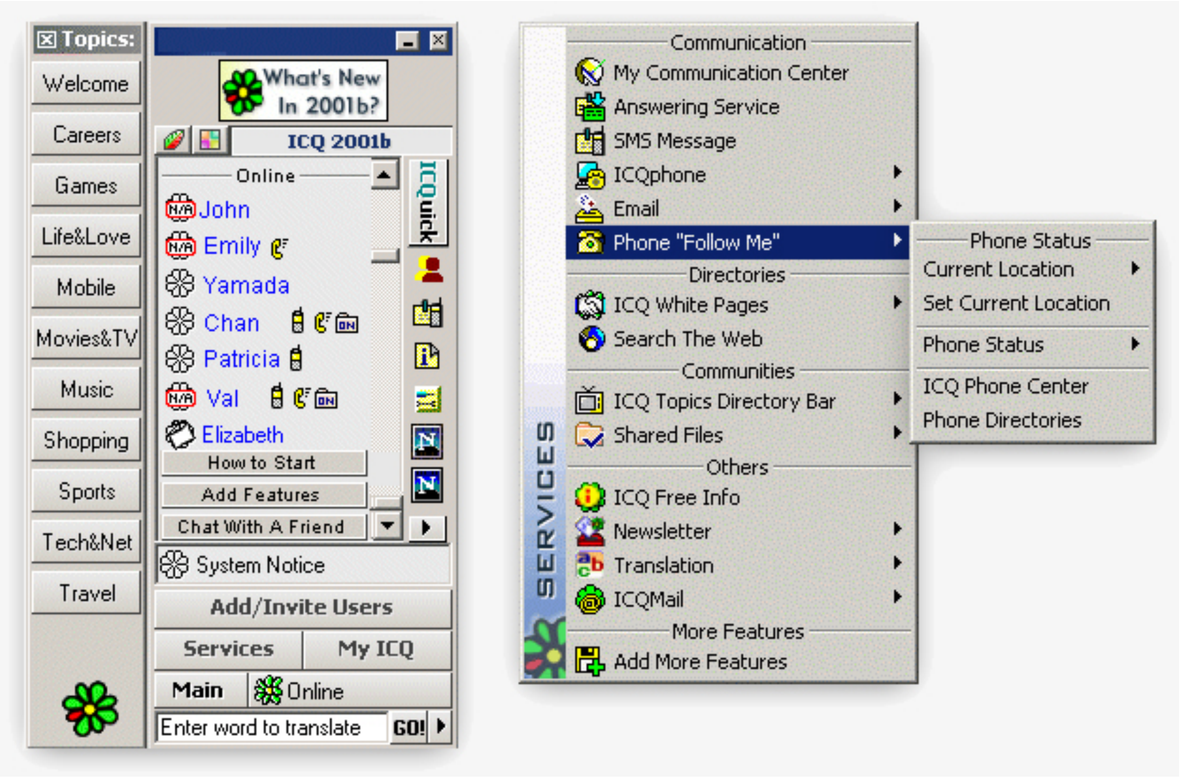
- O TLS revela ainda muita meta informação (tamanho e número de mensagens)
- A análise de tráfego permite muitas vezes reconhecer:
  - com que servidor se está a interagir
  - que operações se estão a fazer
- Isto também se aplica a tráfego transferido via TOR!

Secure Messaging
(extra: não sai para teste)

Secure Messaging

- Ano 2004:
  - Existiam já mecanismos de comunicação por instant messaging, geralmente centralizados
    - AIM, MSN, ICQ, …
  - Começaram a aparecer os primeiros overlays que pretendiam garantir “segurança” contra adversários externos e o próprio servidor.
  - Esses overlays eram baseados nas tecnologias de chave pública dos anos 90: S/MIME, PGP/GPG.

I Seek You (ICQ)



Off the Record Messaging

- Ano 2004: Off the Record Messaging (OTR)
  - Requisitos de segurança e privacidade inovadores:

"*We argue that **not only must encryption be used to hide the contents of the conversation**, but also, the encryption must provide **perfect forward secrecy** to protect from future compromises.*
*Additionally, **authentication must be used** to ensure that the person on the other end is who they claim to be.*
*However, the **authentication mechanism must offer repudiation**, so that the communications remain personal and unverifiable to third parties.*
Only with these properties can privacy similar to real-world social communications be achieved."

Off the Record Messaging

- Mecanismos clássicos não satisfaziam todos os requisitos:
  - Usam assinaturas digitais para autenticação, o que torna as mensagens não repudiáveis
  - Se não usam assinaturas digitais para autenticação são completamente inseguros contra Man-in-the-Middle
  - Muitas vezes, comprometendo uma chave de longa duração, o adversário consegue recuperar todas as mensagens trocadas

Off the Record Messaging

- Ideias chave para perfect-forward-secrecy (*ratcheting*):
  - DH + Assinaturas para handshake inicial
  - DH re-keying sobre autenticação simétrica $k_{ij} = H(g^{x_i y_j})$

$$
\begin{aligned}
A \rightarrow B: &\quad g^{x_1} \\
B \rightarrow A: &\quad g^{y_1} \\
A \rightarrow B: &\quad g^{x_2}, E(M_1, k_{11}) \\
B \rightarrow A: &\quad g^{y_2}, E(M_2, k_{21}) \\
A \rightarrow B: &\quad g^{x_3}, E(M_3, k_{22})
\end{aligned}
$$

Descarta-se logo que possível k_{ij}

Descarta-se logo que possível x, y, g^x e g^y

Off the Record Messaging

- Ideia chave para repúdio:
  - DH + Assinaturas para handshake inicial
  - Alguém que registe todo o trace tem uma prova? => Apenas do handshake
  - A chave de MAC é calculada como $H(K_{enc})$
  - Quando a chave K_{enc} muda, a chave de MAC é divulgada!
  - Porquê => qualquer mensagem autenticada poderia ser feita por qualquer um!

Signal

- Vários protocolos de messaging seguro com “end-to-end-encryption” surgiram nos últimos anos, que protegem contra a “curiosidade” do próprio servidor.

- Os grandes fabricantes de software e fornecedores de serviços adoptaram esse standard de segurança, sob pena de perderem utilizadores.

- O mais proeminente é talvez o Signal, que tem as suas próprias aplicações, e é também adotado pelo Whatshapp e pelo Facebook Messenger.

2E9 utilizadores!!!

Estrutura do Signal

- A comunicação tem de ser possível mesmo com uma das parties off-line
  - Implica utilizar o servidor como buffer e, inicialmente, como canal que autentica utilizadores
- O bootstrap é feito quando um utilizador regista uma identity key (chave DH de longa duração) no servidor
  - autenticação com base no telemóvel

Estrutura do Signal

- O acordo de chaves tem 3 partes
  - Handshake inicial: extended tripple Diffie-Hellman (X3DH)
  - Asymmetric ratchet (quando recebemos DH fresco): recalcula-se chave de sessão com mistura de DH antigo e DH novo
  - Symmetric ratchet (quando não recebemos DH fresco): recalcula-se chave de sessão com base em hashing
- Cada mensagem é protegida com uma chave diferente (perfect forward secrecy)
- Introduz-se um novo objetivo de segurança: post-compromise security, que permite recuperar segurança mesmo se o estado interno do protocolo for revelado.

(Figuras de https://eprint.iacr.org/2016/1013.pdf)

Comunicação Offline

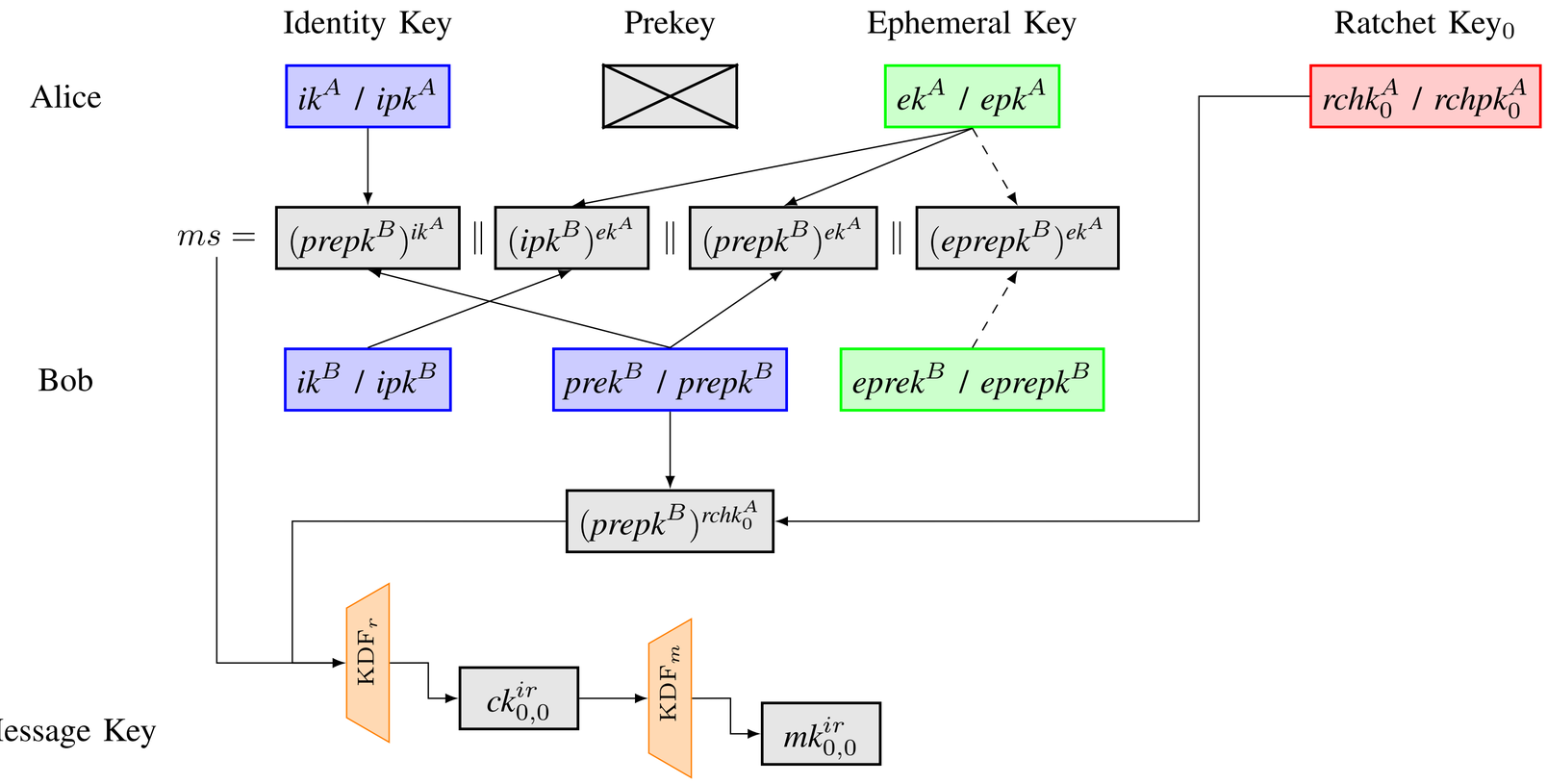
- Para garantir comunicação quando um contacto está offline:
  - todos os contactos submetem regularmente um número significativo de chaves DH efémeras para o servidor.
- Quando queremos enviar uma mensagem a um contacto:
  - podemos pedir uma dessas chaves ao servidor, para termos um elemento fresco para o ratcheting assimétrico ou handshake inicial.

Registo

- Cada recetor regista o seguinte conjunto de chaves públicas DH:
  - Uma identity key idk
  - Uma signed prekey de médio prazo assinada com Kid (partilhadas por todos os emissores)
  - Chaves DH de curto prazo, descartadas pelo servidor uma vez entregues a um emissor
- Em resumo:
  - existem chaves DH de curta e média duração para acautelar ataques MitM localizados no tempo => necessário interceptar ambas
  - existem chaves de assinatura de longa duração => necessário intervir no início para ser possível lançar ataques MiM sobre chaves DH de média duração

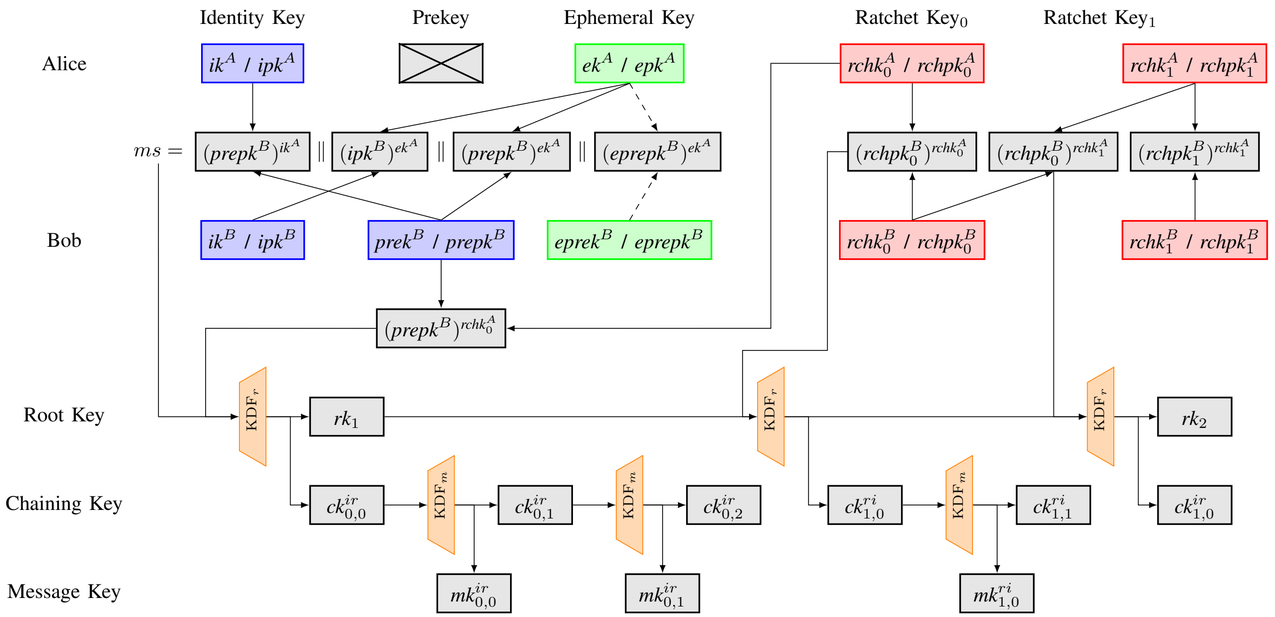
Tripple Handshake

- Perspectiva Alice:
- Mensagem enviada identifica chave efémera utilizada
- Observar como se combinam vários Diffie-Hellman:
  - concatenam-se várias chaves da forma g^{xy}
  - calcula-se o hash (KDFr) => chave de cadeia ck
  - outro hash (KDFm) => chave para uma mensagem mk



Ratcheting

- Depende das mensagens transmitidas e recebidas:
- Quando transmitimos várias mensagens seguidas, usamos ratchet simétrico.
- Quando recebemos um novo valor DH do destinatário, então fazemos ratchet assimétrico.
- Destroem-se as chaves de transmissão (mk) e guarda-se a última chave de cadeia (ck)



- Como gerimos o “esquecimento” do lado do receptor?
- As mensagens podem chegar desordenadas, portanto temos de nos lembrar das mk passadas até recebermos uma mensagem cifrada com essa chave.
- Se avançarmos a chain, todas as mensagens intermédias estão protegidas!

Pressupostos e garantias

- Recebemos o idk inicial por canal autêntico (servidor honest but curious)
- Recebemos DH do primeiro triple handshake por canal autêntico.
- Isto garante confidencialidade e autenticidade do master secret.
- O esquecimento das chaves garante perfect forward secrecy.
- Se houver corrupção total do estado, podemos recuperar segurança (post compromise security) se houver um ratchet assimétrico que o adversário não controla poderemos recuperar segurança (talvez temporariamente)
- Note-se que corrupção total do estado implica que idk já não é autentica, e portanto novas sessões serão vulneráveis a MitM.
- Não tem como objectivo a deniability, mas seria possível usar qualquer coisa como no OTR.