Applied Cryptography
Spring Semester 2023
Lecture 32

Kenny Paterson (@kennyog), Felix Günther

Applied Cryptography Group

https://appliedcrypto.ethz.ch/

Overview of this lecture

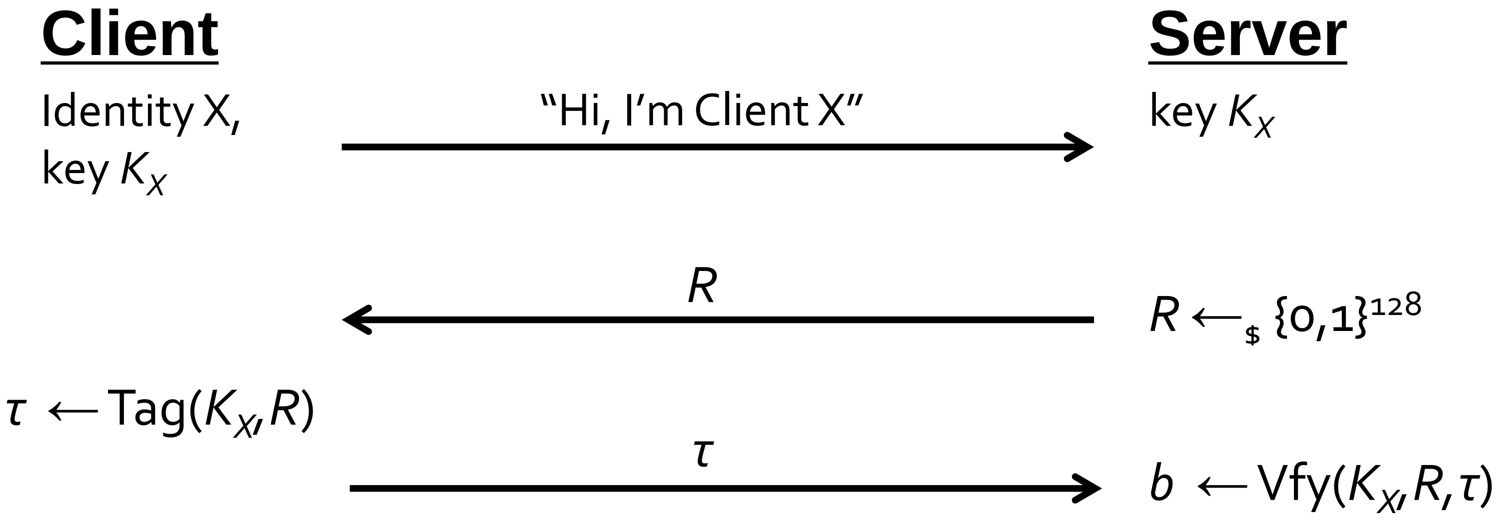
- Entity authentication
- Authenticated Key Exchange
- Extra slides: SIM authentication in GSM/UMTS/5G

Entity authentication

Entity authentication

- An entity authentication protocol provides an assurance – at a particular point in time – about the identity of a communicating partner.
- Built using cryptographic mechanisms in combination with interaction.
- Different to data origin authentication (where did this data come from?) and integrity (has this data been modified?).
- But, clearly, if you have a guarantee about data origin, and can combine it with a guarantee about recency, then you can obtain entity authentication.
- Identity is a slippery concept; we will equate with "demonstrating possession of a key that it is assumed is only known to a particular entity".
- Our focus will be on unilateral entity authentication. Mutually authenticating protocols also exist, built in similar ways.

Entity Authentication from Challenge-Response Protocols: Using a MAC scheme



Client

Identity X,
key K_X

“Hi, I’m Client X”

Server

key K_X

R

$R \leftarrow_{\$} \{0,1\}^{128}$

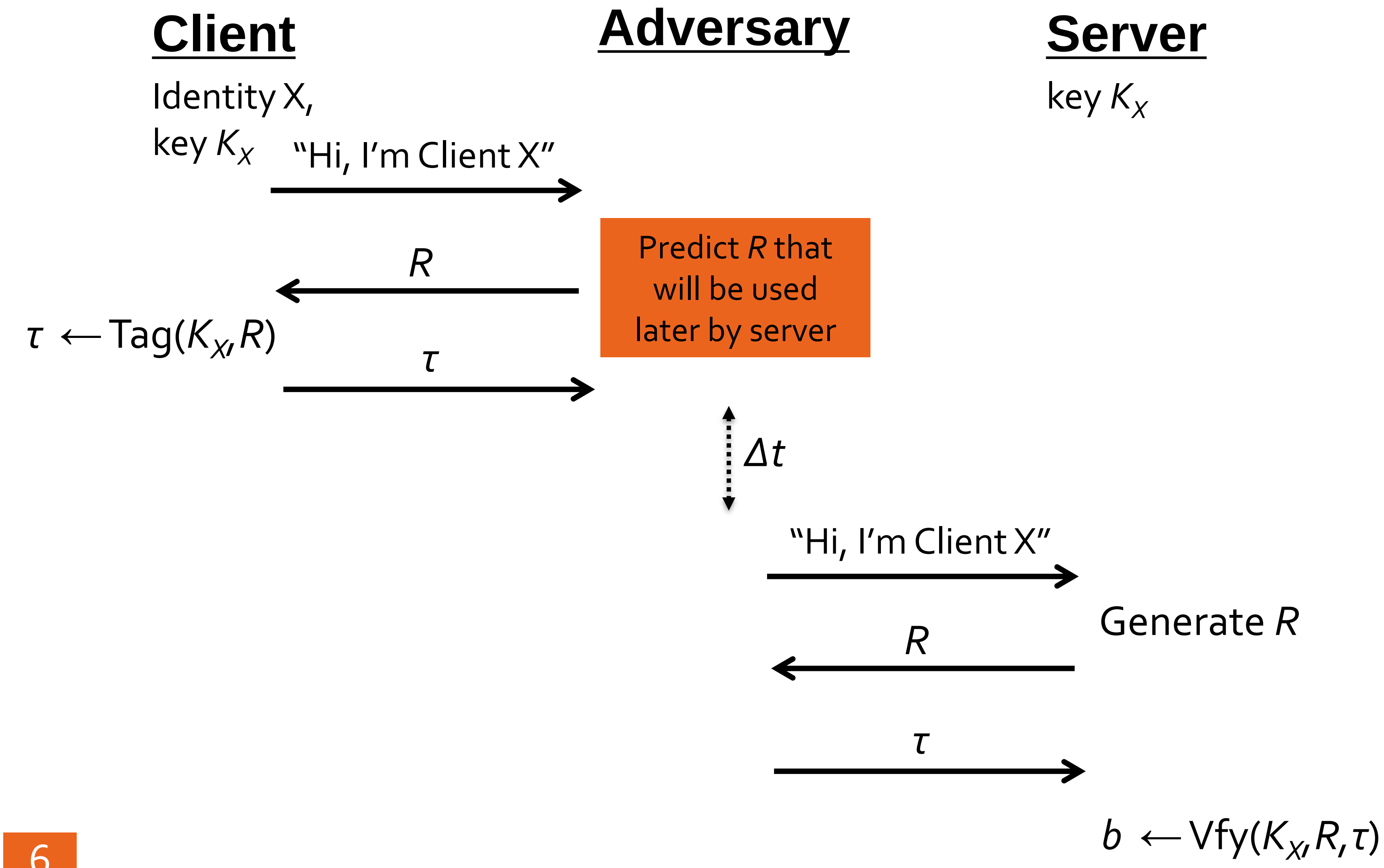
$\tau \leftarrow \mathrm{Tag}(K_X, R)$

τ

$b \leftarrow \mathrm{Vfy}(K_X, R, \tau)$

- Client and server have a shared key for MAC scheme (KGen,Tag, Vfy).
- Server issues random challenge; client can only compute response if it knows key K_X.
- Security relies on MAC unforgeability and unpredictability of challenge.
- Unilateral authentication: server gets assurance it is “talking” to client X.

Entity Authentication from Challenge-Response Protocols: Importance of Unpredictability of Challenges



Client

Identity X,
key K_X

Adversary

Server

key K_X

"Hi, I'm Client X"

R

Predict R that will be used later by server

$\tau \leftarrow \text{Tag}(K_X, R)$

τ

Δt

"Hi, I'm Client X"

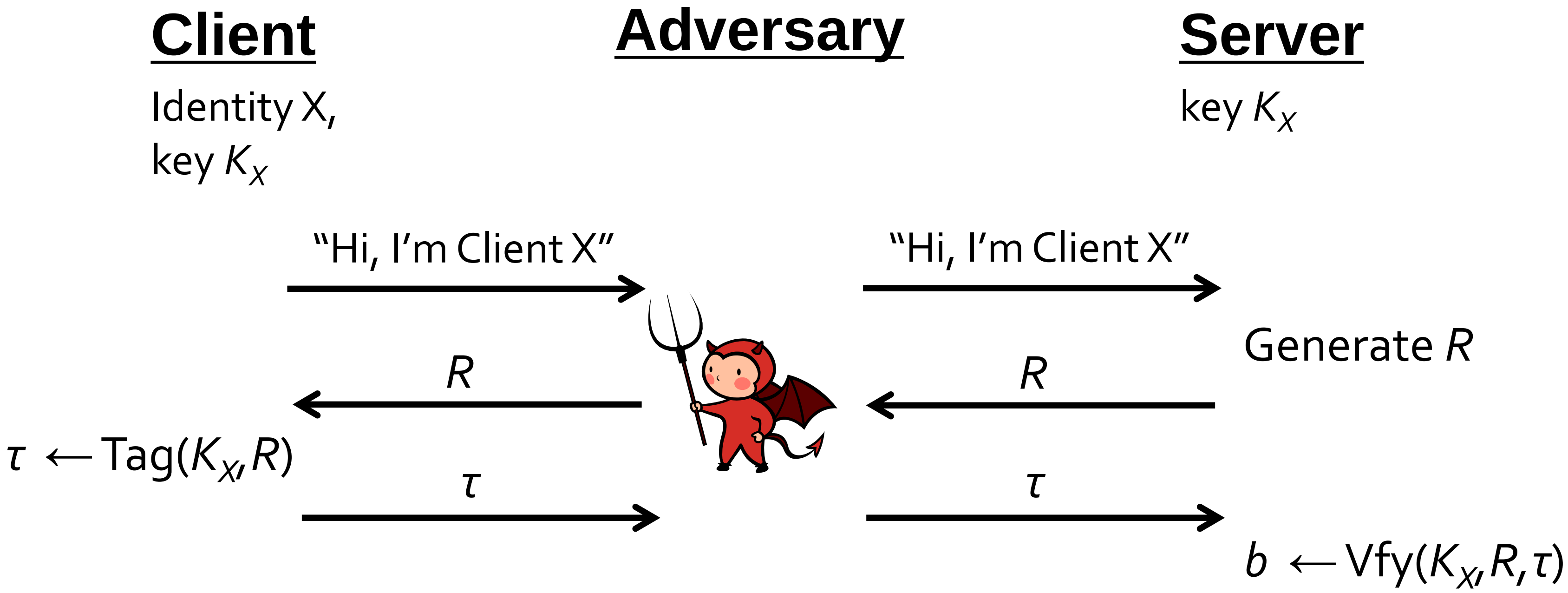
Generate R

R

τ

$b \leftarrow \text{Vfy}(K_X, R, \tau)$

Entity Authentication from Challenge-Response Protocols: Attack or Not?



Client

Identity X,
key K_X

Adversary

Server

key K_X

"Hi, I'm Client X"

"Hi, I'm Client X"

Generate R

R

R

$\tau \leftarrow \mathrm{Tag}(K_X,R)$

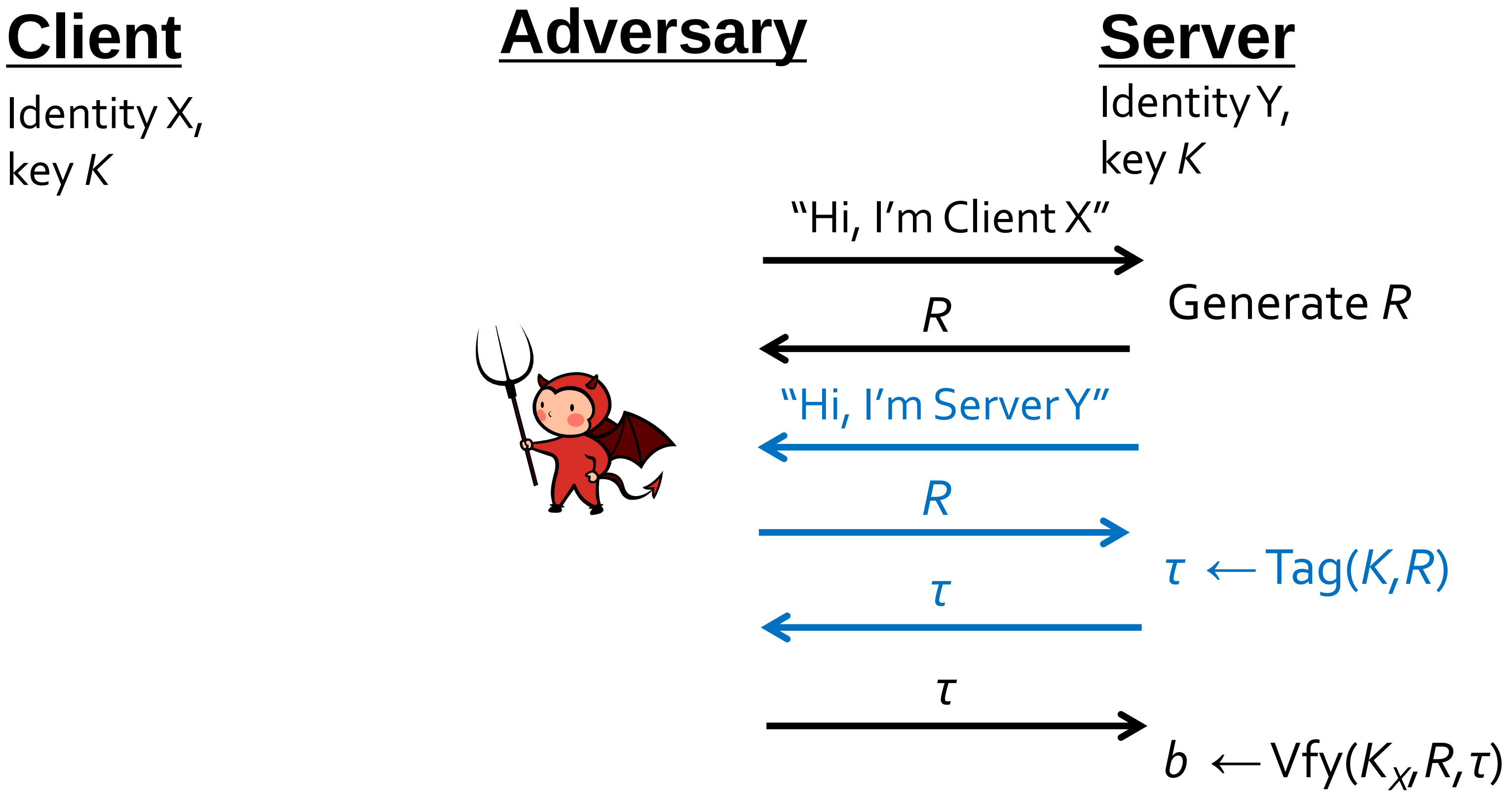
τ

τ

$b \leftarrow \mathrm{Vfy}(K_X,R,\tau)$

- Attacker relays message from client to server and server to client.
- Is this an attack on the authentication property?

Entity Authentication from Challenge-Response Protocols: Reflection Attack



Client

Identity X,
key K

Adversary

Server

Identity Y,
key K

"Hi, I'm Client X" →

Generate R

R ←

"Hi, I'm Server Y" ←

R →

$\tau \leftarrow \mathrm{Tag}(K,R)$

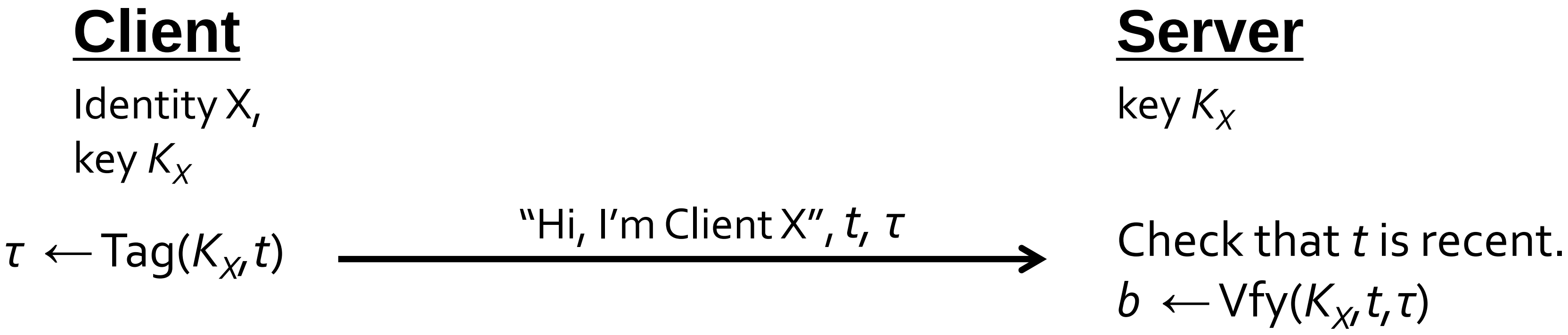
τ ←

τ →

$b \leftarrow \mathrm{Vfy}(K_X,R,\tau)$

- Assumption is that same key K is used to authenticate both parties
- Server begins a second protocol run with adversary in middle of first run.
- Attack illustrates danger of violating key separation principle OR the need for identifiers in messages that are integrity protected.

Entity Authentication from Challenge-Response Protocols: Using Timestamps



Client

Identity X,
key K_X

$\tau \leftarrow \mathrm{Tag}(K_X, t)$

"Hi, I'm Client X", t, τ $\longrightarrow$

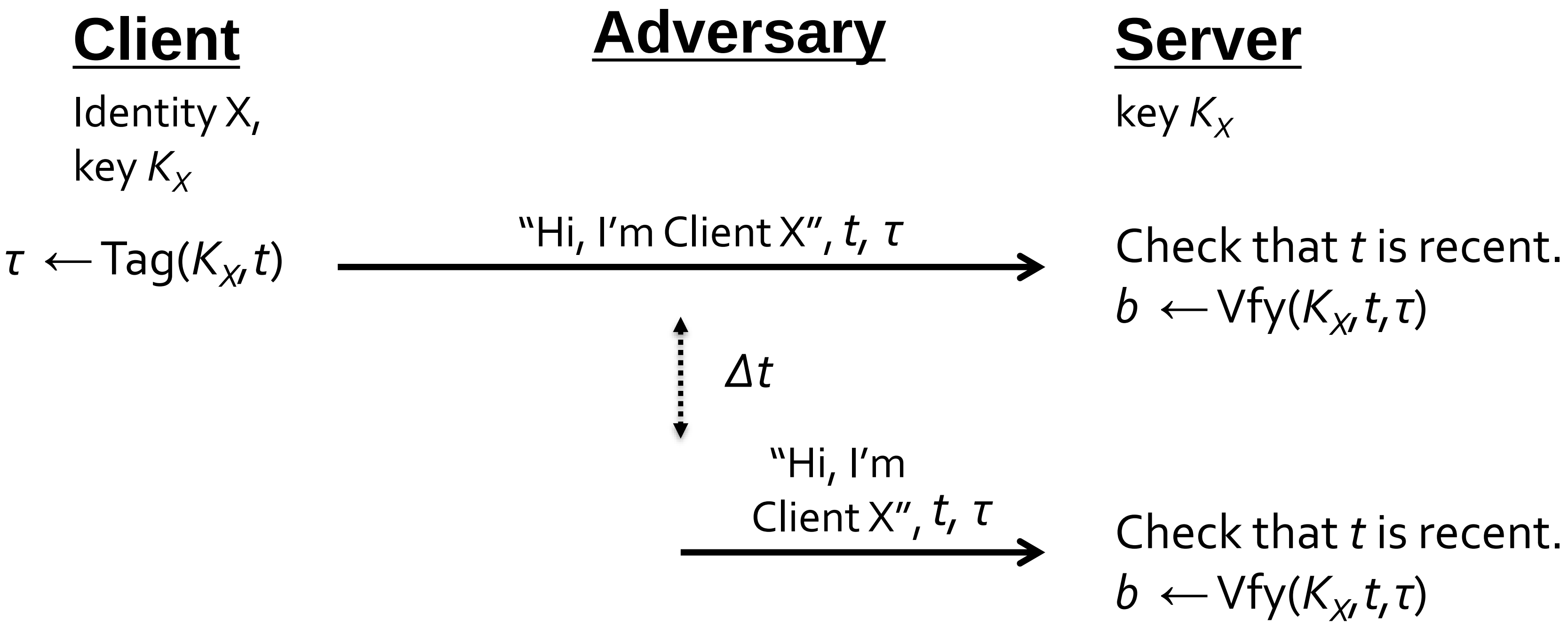
Server

key K_X

Check that t is recent.
$b \leftarrow \mathrm{Vfy}(K_X, t, \tau)$

- Client and server have a shared key and a roughly synchronised clock; t denotes the clock time.
- Client can only compute message if it knows key K_X.
- Server needs to check that t is "recent" but allow a delta for network latency.
- Security relies on MAC unforgeability and unpredictability of challenge.
- Unilateral authentication: server gets assurance it is "talking" to client X.

Entity Authentication from Challenge-Response Protocols: Using Timestamps



Client

Identity X,
key K_X

$\tau \leftarrow \text{Tag}(K_X, t)$

Adversary

"Hi, I'm Client X", t, τ

Δt

"Hi, I'm Client X", t, τ

Server

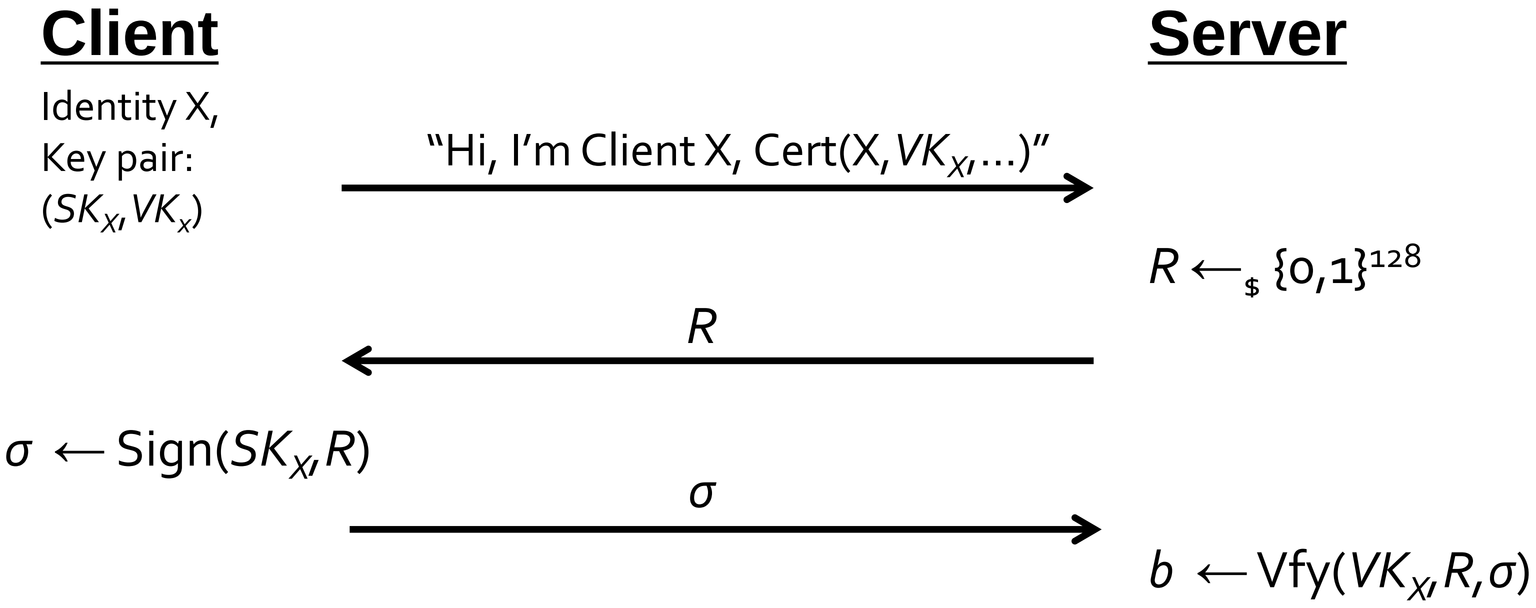
key K_X

Check that t is recent.
$b \leftarrow \text{Vfy}(K_X, t, \tau)$

Check that t is recent.
$b \leftarrow \text{Vfy}(K_X, t, \tau)$

- Adversary can replay message within time Δt and it will be accepted by server assuming it is considered recent enough.
- Attack can be prevented if server keeps a log of recently-received messages.

Entity Authentication from Challenge-Response Protocols: Using a Signature Scheme



Client

Identity X,
Key pair:
(SK_X, VK_X)

Server

"Hi, I'm Client X, Cert(X, VK_X,…)"

$R \leftarrow_{\$} \{0,1\}^{128}$

R

$\sigma \leftarrow \text{Sign}(SK_X, R)$

σ

$b \leftarrow \text{Vfy}(VK_X, R, \sigma)$

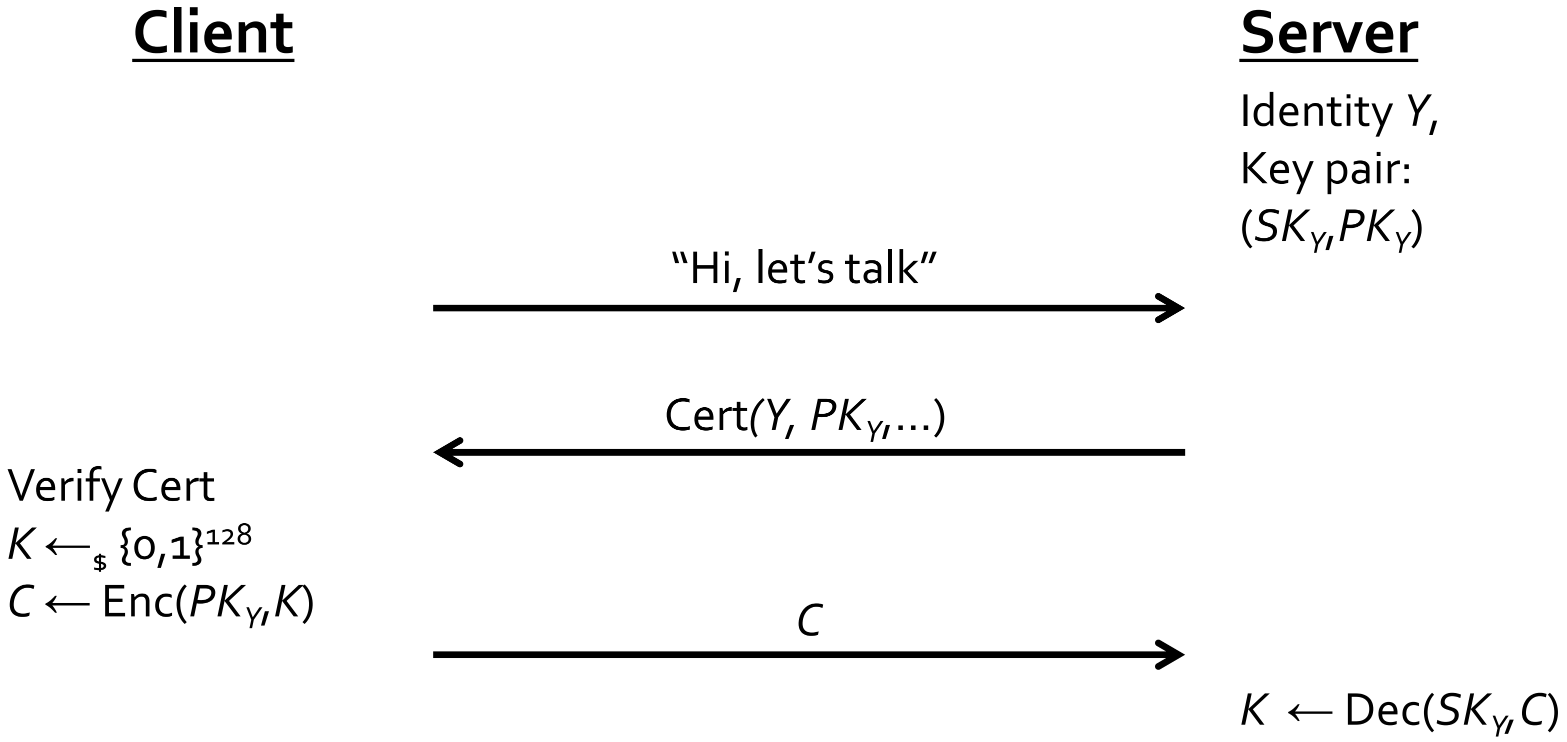
- Server obtains VK_X, Client X's verification key plus certificate (chain).
- Server validates certificate (chain) and extracts VK_X.
- Server issues random challenge; client can only compute response if it knows signing key SK_X.
- Security relies on signature unforgeability and unpredictability of challenge.
- Unilateral authentication: server gets assurance it is "talking" to client X.

Authenticated Key Exchange/Establishment

Authenticated Key Exchange/Establishment

- Our focus so far has been on (unilateral) entity authentication.
- But authentication on its own is rarely useful.
- Authenticated Key Exchange/Establishment (AKE) aims to, beyond authentication, also distribute key material, such that:
  - One or both parties get an assurance about with whom they have established the key material.
  - No attacker can learn anything about the key material.
  - Even if the attacker completely controls the network, can start protocol runs between pairs of parties, ….: the Dolev-Yao adversary.
- Widely used in secure communications protocols as a precursor to bulk data exchange using symmetric techniques.
- In lectures ahead, we will see how this is done in SSL/TLS, Signal, …

Authenticated Key Establishment from PKE



Client

Server

Identity Y,
Key pair:
(SK_Y, PK_Y)

"Hi, let's talk"

Cert$(Y, PK_Y, \ldots)$

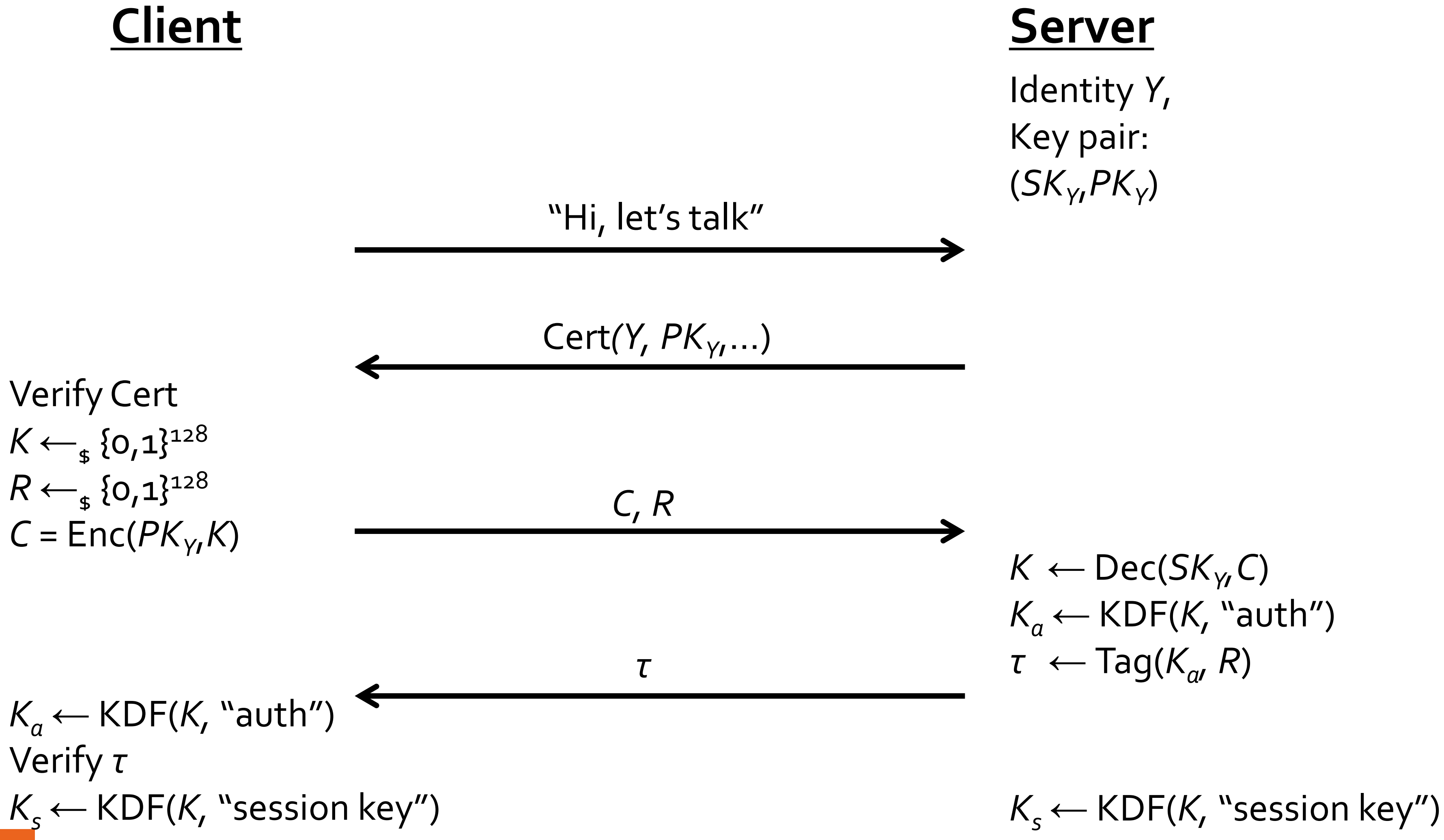
Verify Cert
$K \leftarrow_{\$} \{0,1\}^{128}$
$C \leftarrow \mathrm{Enc}(PK_Y, K)$

C

$K \leftarrow \mathrm{Dec}(SK_Y, C)$

- Client is unauthenticated/anonymous.
- Client selects session key K and encrypts to public key of server (after verifying server certificate).
- Is the server authenticated to the client in this protocol?

Authenticated Key Establishment from PKE: Adding Explicit Server Authentication



Client

Server

Identity Y,
Key pair:
(SK_Y, PK_Y)

“Hi, let’s talk” →

← Cert(Y, PK_Y,…)

Verify Cert
$K \leftarrow_{\$} \{0,1\}^{128}$
$R \leftarrow_{\$} \{0,1\}^{128}$
$C = \mathrm{Enc}(PK_Y, K)$

C, R →

$K \leftarrow \mathrm{Dec}(SK_Y, C)$
$K_a \leftarrow \mathrm{KDF}(K, \text{“auth”})$
$\tau \leftarrow \mathrm{Tag}(K_a, R)$

← τ

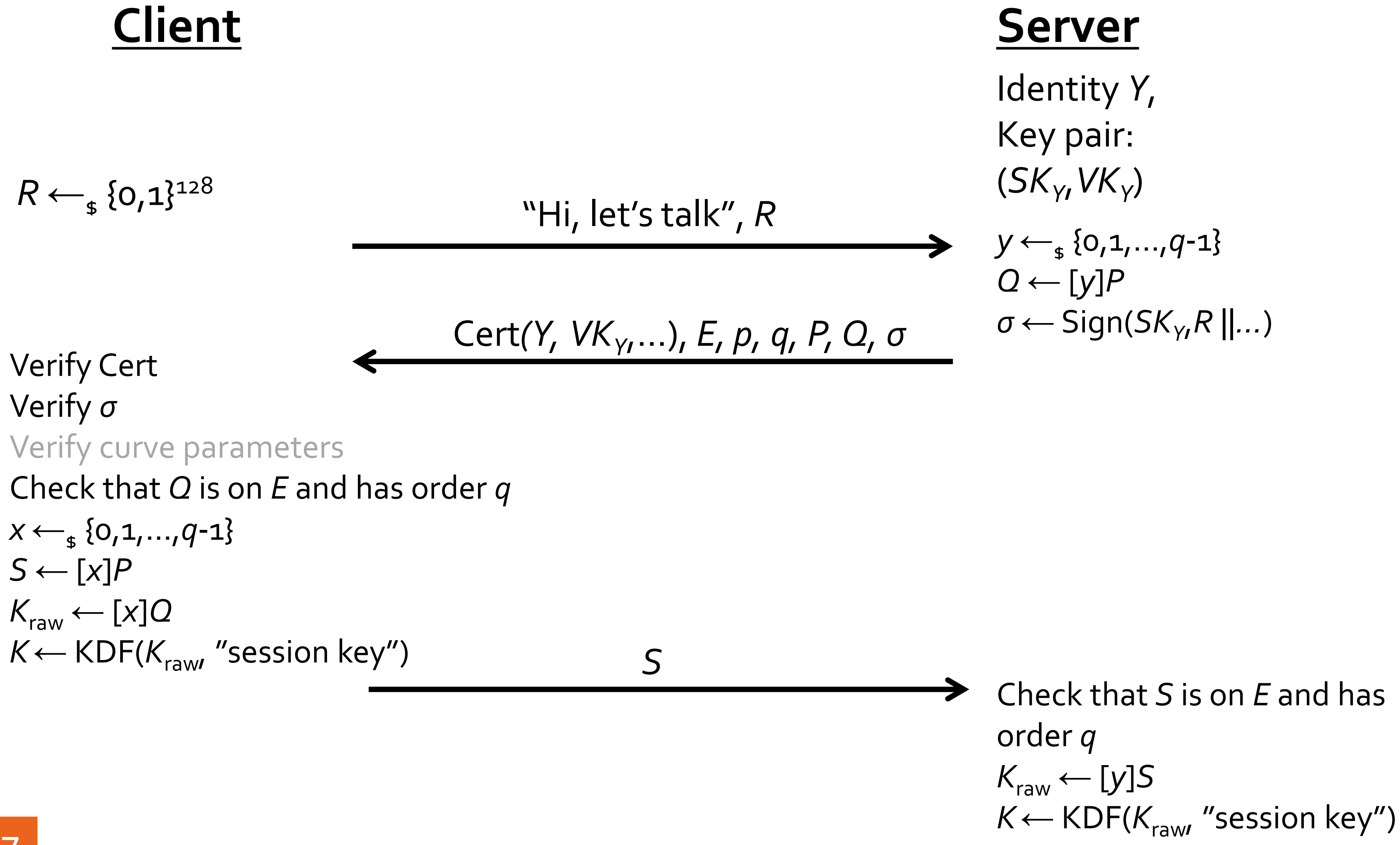
$K_a \leftarrow \mathrm{KDF}(K, \text{“auth”})$
Verify τ
$K_s \leftarrow \mathrm{KDF}(K, \text{“session key”})$

$K_s \leftarrow \mathrm{KDF}(K, \text{“session key”})$

Forward Security

- The preceding protocol is close to the SSL/TLS Handshake Protocol up to and including TLS 1.2, when RSA encryption is used for key transport.
- What happens to security of session keys if the long-term server key SK_Y is compromised?
  - Clearly an active attacker can then impersonate the server to clients, and a passive attacker can learn any new session keys.
  - But a passive attacker who recorded **old** protocol runs can also recover the corresponding session keys.
- This PKE-based protocol **fails** to provide **forward security**: the property that compromise of a long-term key should **not** lead to the insecurity of previously established session keys.
- NB forward security refers to the security of session keys from the past!

Authenticated Key Exchange from Elliptic Curve Diffie-Hellman Ephemeral and Signatures



Client

Server

Identity Y,
Key pair:
(SK_Y, VK_Y)

$R \leftarrow_{\$} \{0,1\}^{128}$

"Hi, let's talk", R →

$y \leftarrow_{\$} \{0,1,\ldots,q\text{-}1\}$
$Q \leftarrow [y]P$
$\sigma \leftarrow \text{Sign}(SK_Y, R \,\|\ldots)$

← Cert(Y, VK_Y,…), E, p, q, P, Q, σ

Verify Cert
Verify σ
Verify curve parameters
Check that Q is on E and has order q
$x \leftarrow_{\$} \{0,1,\ldots,q\text{-}1\}$
$S \leftarrow [x]P$
$K_{raw} \leftarrow [x]Q$
$K \leftarrow \text{KDF}(K_{raw}, \text{"session key"})$

S →

Check that S is on E and has order q
$K_{raw} \leftarrow [y]S$
$K \leftarrow \text{KDF}(K_{raw}, \text{"session key"})$

Authenticated Key Exchange from Diffie-Hellman and Signatures – Notes

- Server selects curve parameters (E, p, q, P).
- Client should verify them, but this is complex and expensive to do.
- It is more common to use a standardised curve, and encode the choice in the first message sent by server.
- Server signature includes client's chosen random value R.
- This provides challenge-response authentication of server to client.
- $S \leftarrow [x]P$ and $Q \leftarrow [y]P$ are *ephemeral* ECDH values: one-time use.
- Client is anonymous/unauthenticated; client authentication can be added via signatures too.
- Client can compute session key K after receipt of server's ECDH value, so could already send encrypted data on its second flow.

Authenticated Key Exchange from Diffie-Hellman and Signatures – Forward Security

- The preceding protocol is close to the SSL/TLS Handshake Protocol up to TLS 1.2 when "DH + signatures" is used for key establishment.
- A significant omission is the inclusion of MAC values via "ClientFinished" and "ServerFinished" messages.
- These are needed to prevent a certain class of attack called an **Unknown Keyshare Attack**.
- Now what happens to security of session keys if the long-term server key SK_Y is compromised?
- Clearly an **active** attacker can still impersonate the server to clients.
- But a **passive** attacker cannot recover any new session keys.
- And a passive attacker who recorded **old** protocol runs can no longer recover the corresponding session keys.
- So we (informally) obtain **forward security**.

Authenticated Key Exchange from Diffie-Hellman - II

Alice

Identity IDX, ECDH
key pair: $(a, A = [a]P)$

$x \leftarrow_{\$} \{0,1,\ldots,q\text{-}1\}$
$X \leftarrow [x]P$

$\xrightarrow{\text{Cert}(IDX, A, \ldots),\ X}$

Bob

Identity IDY, ECDH
key pair: $(b, B = [b]P)$

Check Cert
Check that X is on E
$y \leftarrow_{\$} \{0,1,\ldots,q\text{-}1\}$
$Y \leftarrow [y]P$
$K_{raw} \leftarrow [b]A \,||\, [y]X$
$K \leftarrow \text{KDF}(K_{raw}, \text{“sk”})$

$\xleftarrow{\text{Cert}(IDY, B, \ldots),\ Y}$

Check Cert
Check that Y is on E
$K_{raw} \leftarrow [a]B \,||\, [x]Y$
$K \leftarrow \text{KDF}(K_{raw}, \text{“sk”})$

Authenticated Key Exchange from Diffie-Hellman - II

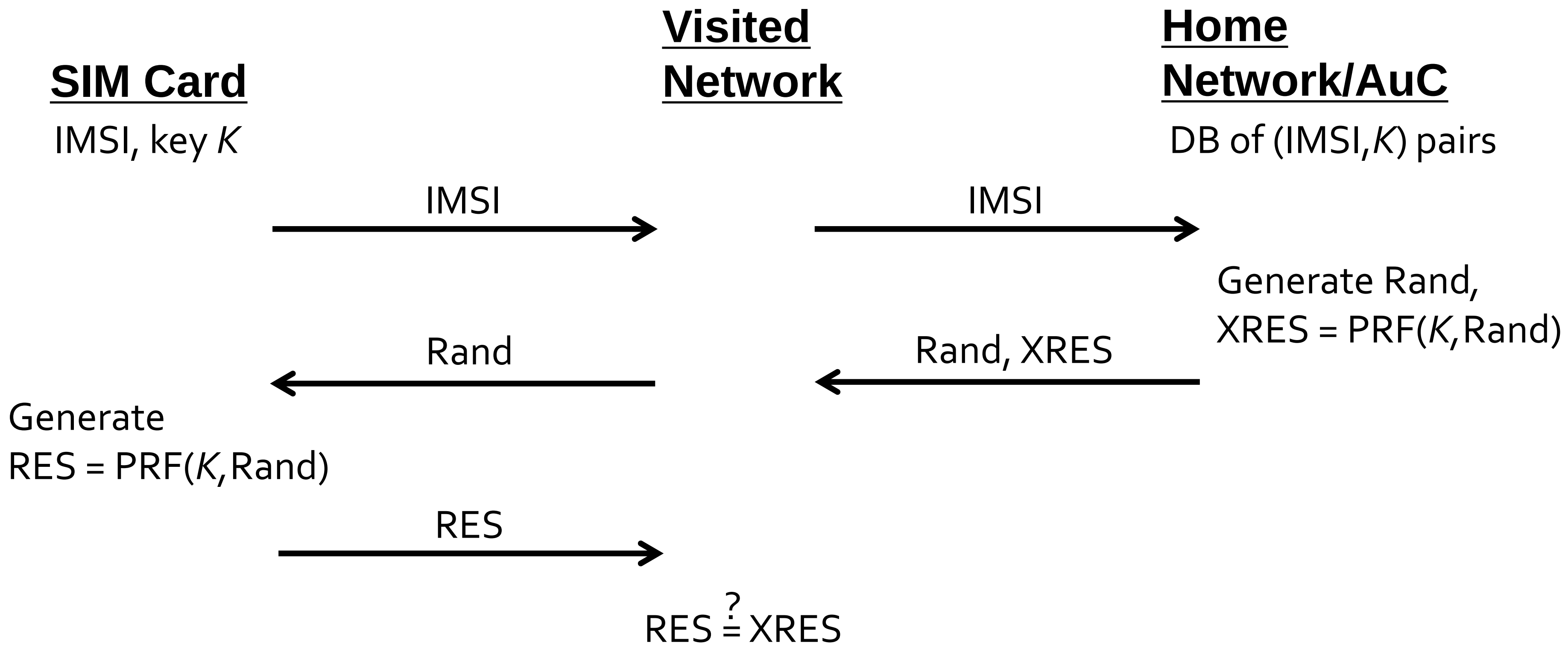
- The preceding protocol is (historically) called the **Unified Model** protocol.
- A short-term DH exchange is authenticated using a long-term DH key; authentication here is only implicit.
- This protocol achieves forward security.
- But it fails to achieve security against **Key Compromise Impersonation** (KCI) attacks: an attacker who learns Alice’s long-term key can impersonate other parties **to** Alice (as well as impersonating Alice to other parties).
- The idea of combining long-term and short-term DH values is used extensively in the Signal protocol.

Formal Security Analysis of AKE

- So far we only **informally** discussed security for entity authentication and AKE protocols.
- Many early protocols were designed without formal security models/definitions; many attacks were found.
- The field has gradually been made rigorous, starting with Bellare-Rogaway’93.
- Complexity arises from interactivity of protocols and an inherently multi-party setting.
- Complexity also arises from many attack types: on forward security, UKS attacks, KCI attacks,… and many different security goals.
- Real protocols (e.g. TLS) negotiate version, algorithms, authentication modes, etc., as part of the protocol, making things even more complex.
- Historically, there has been a large gap between real protocols and what can be formally modelled in a tractable way.
- The gap has closed significantly, we’ll see examples in the lectures ahead.

Extra slides on
Entity Authentication in GSM/UMTS/5G

Entity Authentication in GSM/UMTS/5G (Simplified)



SIM Card

IMSI, key *K*

Visited Network

Home Network/AuC

DB of (IMSI,*K*) pairs

IMSI

IMSI

Generate Rand,
XRES = PRF(*K*,Rand)

Rand

Rand, XRES

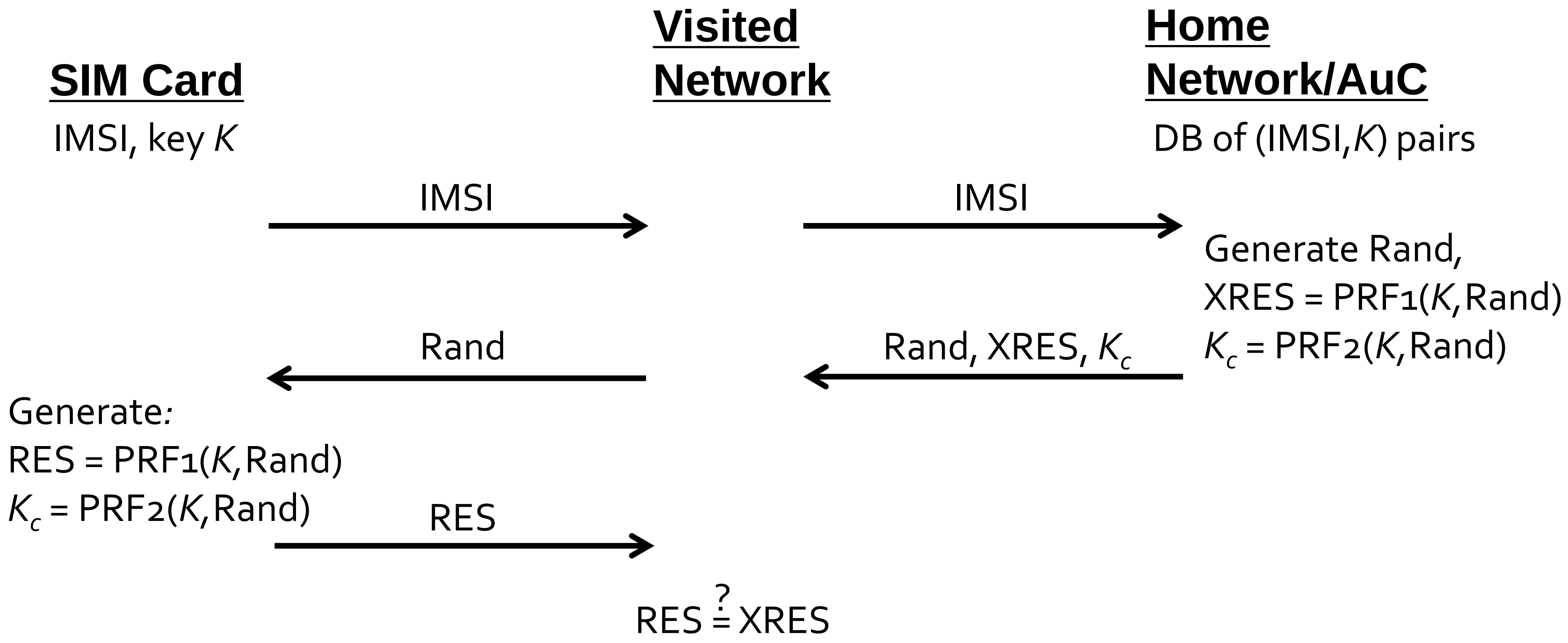
Generate
RES = PRF(*K*,Rand)

RES

RES $\overset{?}{=}$ XRES

- AuC = Authentication centre.
- Keys *K* emplaced in SIM Card during/after manufacture, copy sent to AuC.
- Only authentication of SIM Card to Visited Network shown; UMTS and 5G also authenticate network to SIM Card (not shown).
- This authentication scheme allows roaming between networks, provided home network has agreement with visited network.

Entity Authentication and Key Establishment in GSM/UMTS/5G (Simplified)



SIM Card

IMSI, key K

Visited Network

Home Network/AuC

DB of (IMSI,K) pairs

Generate Rand,
$XRES = PRF1(K,Rand)$
$K_c = PRF2(K,Rand)$

Generate:
$RES = PRF1(K,Rand)$
$K_c = PRF2(K,Rand)$

$RES \overset{?}{=} XRES$

- Encryption key K_c derived using a second PRF
- Encryption using a stream cipher between mobile equipment containing SIM Card and base station of visited network.
- Encryption is not end-to-end, but only on wireless portion.
- System has in-built facility for legal interception.