ca technologies

Trust in TLS

Tomáš Pavelka – May 2018 – 5.08

ca technologies

For Informational Purposes Only

This presentation was based on current information and resource allocations as of **May 2018** and is subject to change or withdrawal by CA at any time without notice. Not withstanding anything in this presentation to the contrary, this presentation shall not serve to (i) affect the rights and/or obligations of CA or its licensees under any existing or future written license agreement or services agreement relating to any CA software product; or (ii) amend any product documentation or specifications for any CA software product. The development, release and timing of any features or functionality described in this presentation remain at CA's sole discretion. Notwithstanding anything in this presentation to the contrary, upon the general availability of any future CA product release referenced in this presentation, CA will make such release available (i) for sale to new licensees of such product; and (ii) to existing licensees of such product on a when and if-available basis as part of CA maintenance and support, and in the form of a regularly scheduled major product release. Such releases may be made available to current licensees of such product who are current subscribers to CA maintenance and support on a when and if-available basis. In the event of a conflict between the terms of this paragraph and any other information contained in this presentation, the terms of this paragraph shall govern.

Certain information in this presentation may outline CA's general product direction. All information in this presentation is for your informational purposes only and may not be incorporated into any contract. CA assumes no responsibility for the accuracy or completeness of the information. To the extent permitted by applicable law, CA provides this presentation "as is" without warranty of any kind, including without limitation, any implied warranties or merchantability, fitness for a particular purpose, or non-infringement. In no event will CA be liable for any loss or damage, direct or indirect, from the use of this document, including, without limitation, lost profits, lost investment, business interruption, goodwill, or lost data, even if CA is expressly advised in advance of the possibility of such damages. CA confidential and proprietary. No unauthorized copying or distribution permitted.

ca technologies

Motivation

- Everybody wants their network communication secured by TLS
- Just turn it on!
- We need certificates, they need to be signed by a trusted Certificate Authority…
- And if we make a mistake, network communication does not work at all.
- And it is really hard to debug
- Or, we have introduced a security hole
- Everything is more difficult if you are on a private network not connected to the internet

ca technologies

Without understanding the “how” it is very hard to set up

It is easier to understand “how” if you know “why”

Why? Bootstrapping Trust

- How do I trust someone I have never met?
- How do I make it scale?
- At the end of this presentation, you should understand how trust is established in TLS and how your actions affect who you trust.

ca technologies

How To Explain TLS?

- Design a TLS-like protocol in steps where each step is susceptible to an attack
- Show how to mitigate the attack and explain why each piece is needed.
- Note this is not how real TLS works. The protocol simplifies things to only show problems related to certificate management.

Actors

- Client – Requests services, sends messages to server, starts conversation
- Server – Provides services, sends messages to client
- Attacker – Can intercept and modify messages sent between client and server.

Goals: Confidentiality, Integrity and Authenticity

Cryptographic protocol modifies messages between client and server so that they satisfy the following goals:

- **Confidentiality** – Attacker cannot read conversation between client and server.
- **Integrity** – Attacker cannot modify conversation between client and server (without client and server noticing).
- **Authenticity** – Attacker cannot pretend to be client or server.

ca technologies

Cryptography 1

Symmetric cipher – only those who have the key can read encrypted data

- SC (data, key) -> encrypted
- SC (encrypted, key) -> data

Asymmetric Cipher – those with public key can read data encrypted with private key

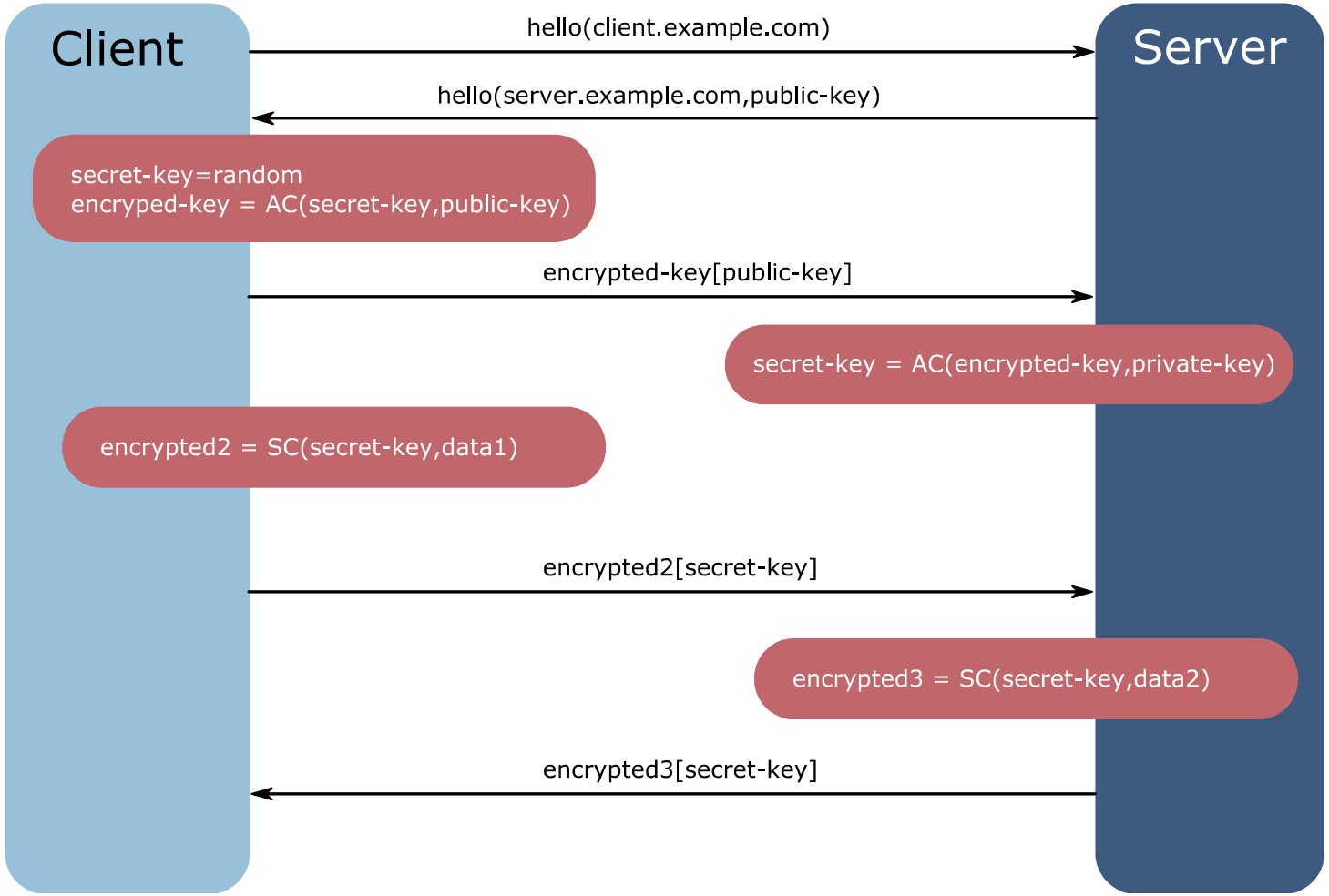
- AC (data, private-key) -> encrypted
- AC (encrypted, public-key) -> data

those with private key can read data encrypted with public key

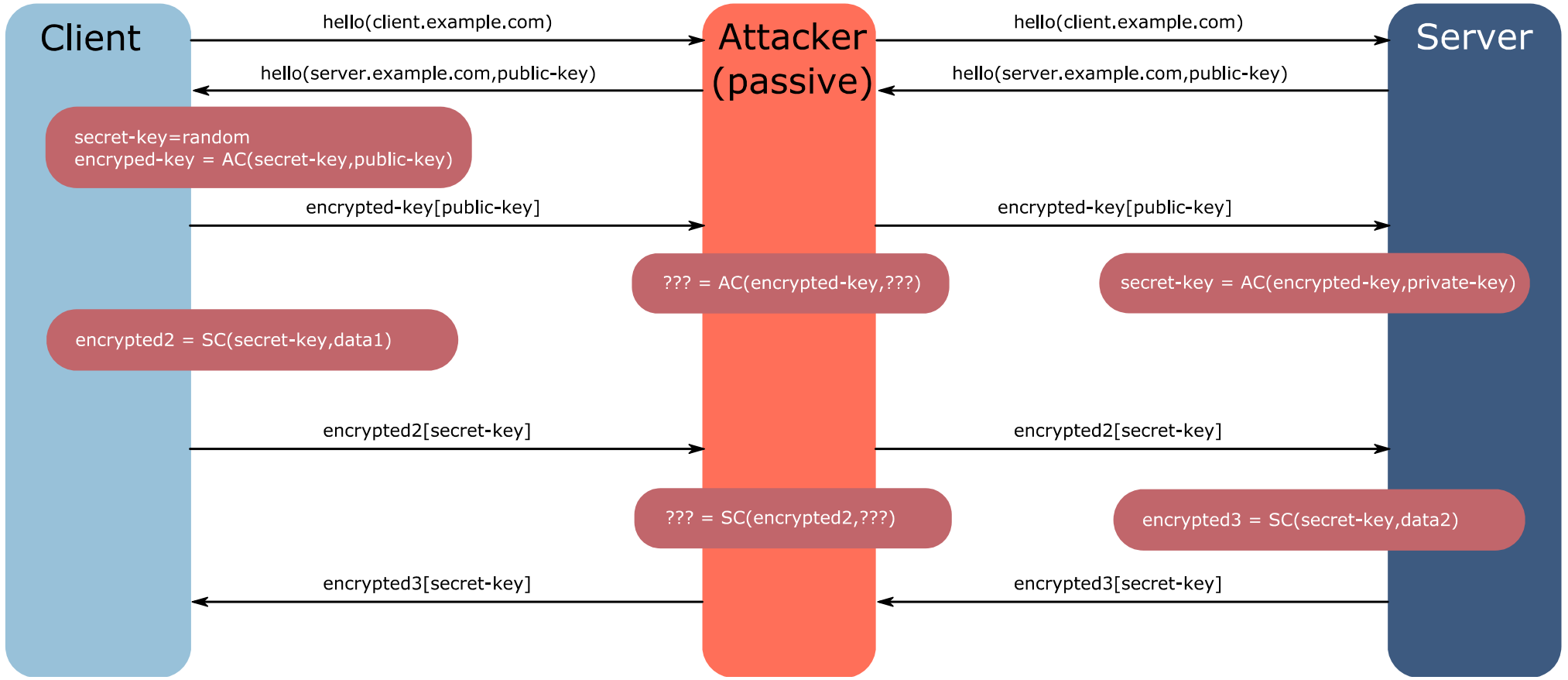
- AC (data, public-key) -> encrypted
- AC (encrypted, private-key) -> data

ca technologies

Use Asymmetric Cipher to Exchange the key for Symmetric Cipher



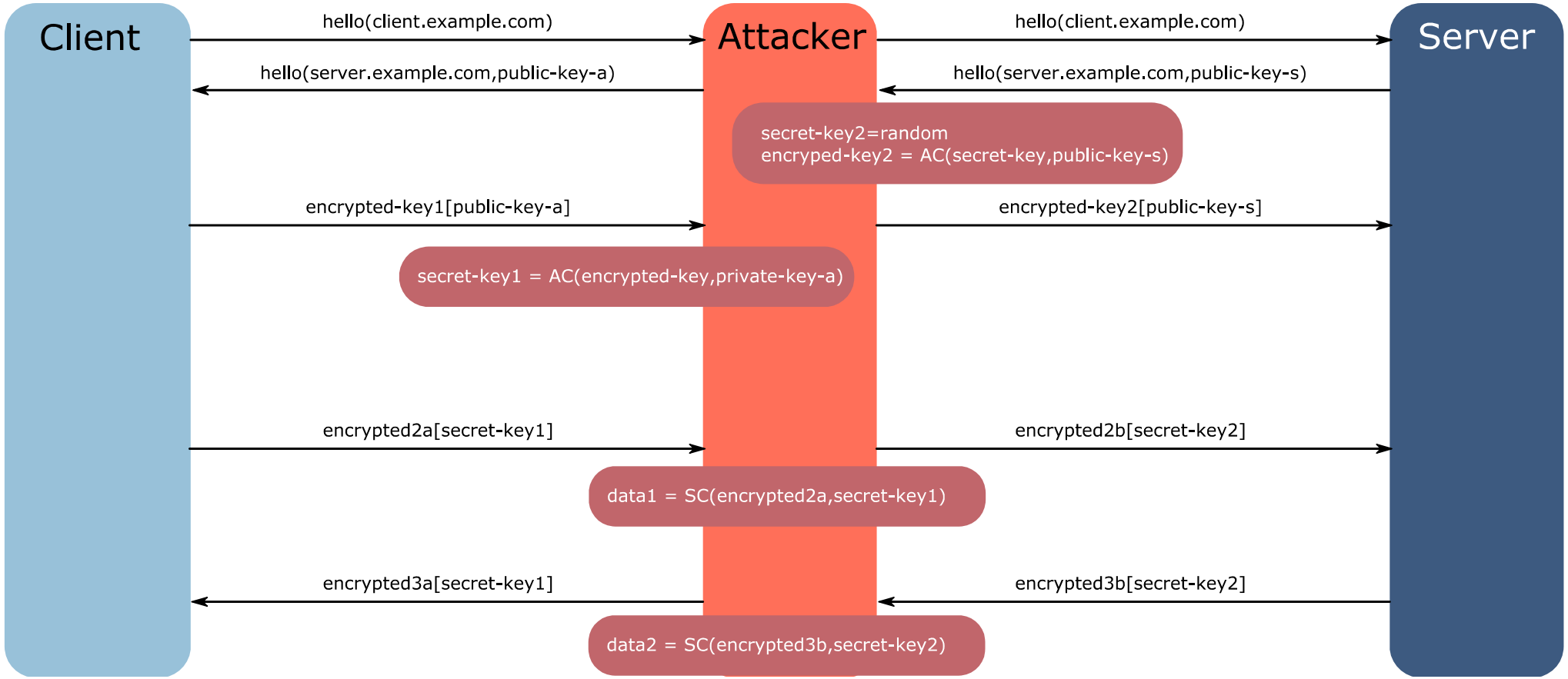
Passive Attacker (Can Only Listen)



Passive Attacker

- The simple setup prevents passive eavesdropping
- No certificates, certificate authorities, no need to configure anything
- Unfortunately, the web does not work this way…

Active Attacker: Man in the Middle Attack



Active Attacker

- IP addresses and host names can be spoofed and an active attacker can modify messages -> our protocol does not provide authenticity -> the attacker can pretend to be the server
- When we lose authenticity, we lose integrity – our messages can be modified on the way without us knowing.
- We also lose confidentiality, the attacker can read our messages.

Cryptography 2

Cryptographic Hash – unfeasible to make hash of one piece of data equal the hash of another piece of data

- CH(data) -> h where size(h) < size(data)

Cryptographic Signature – when you have the public key, you can verify that the data was created by someone who owns the private key

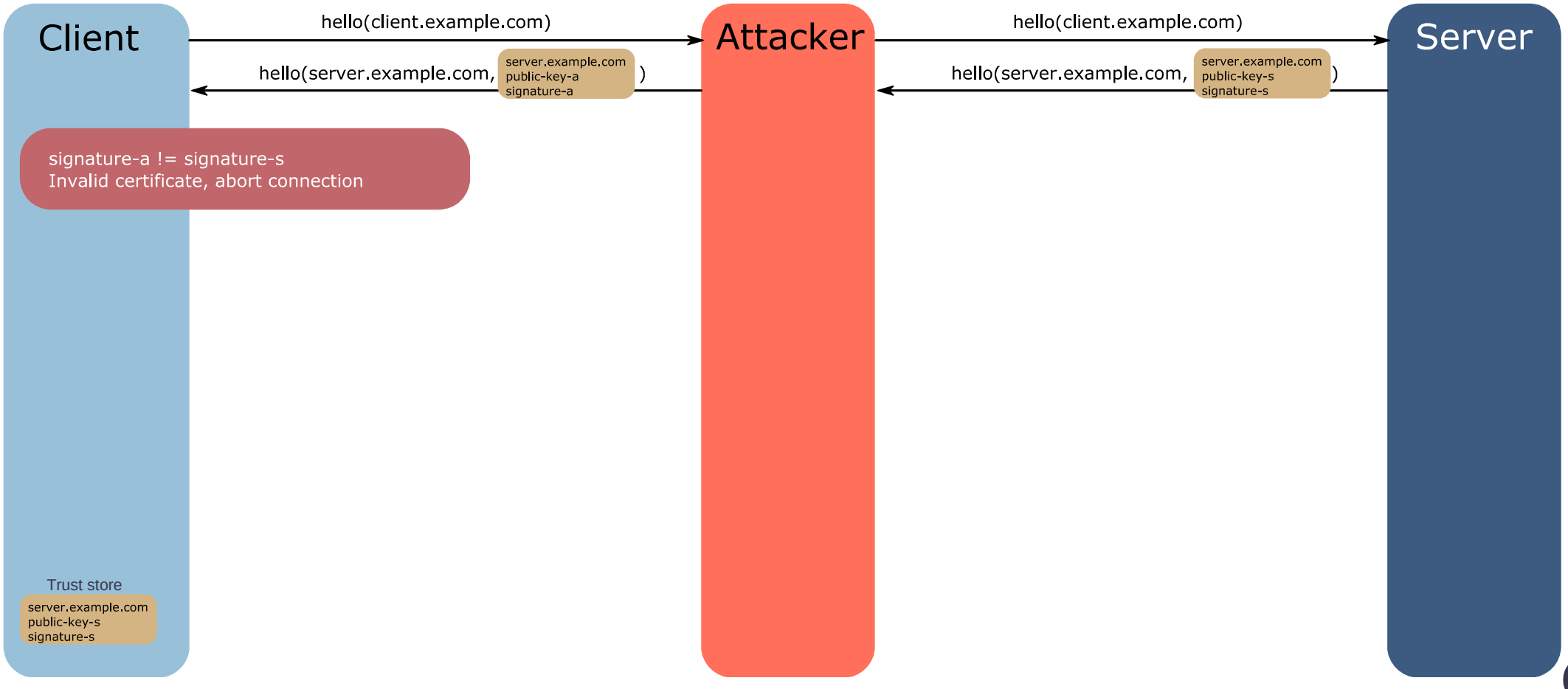
- CS(data, private-key) -> s = AC(CH(data), private-key) -> s

ca technologies

Certificate

- Certificate = some metadata describing the identity of the server + the server's public key + a cryptographic signature
- Trust store – a place where the client keeps all the certificates it trusts
- If client has the certificate of the server in its trust store, the attacker cannot pretend to be the server, because the attacker does not have the server's private key

Certificate Prevents MITM



Scale

- Per http://www.internetlivestats.com/total-number-of-websites/, at the time this presentation was written, there were 1,865,766,475 websites in the world.
- A new website appears just about every second.
- How do you get all these certificates into the client's trust store in a secure way?

Trust Enables Authenticity at Scale

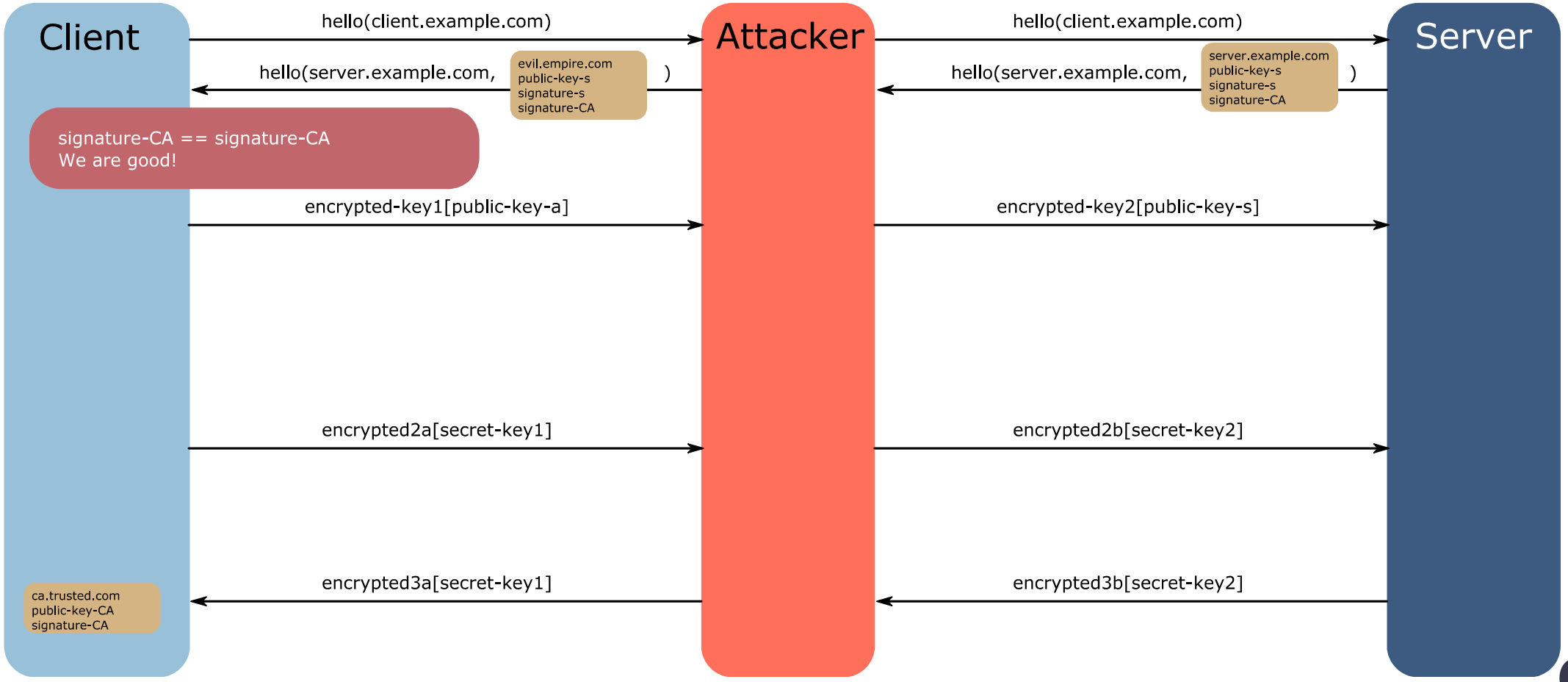
- Small number of trusted third parties vouch for others
- Processes for ensuring the trusted third parties are in fact trustworthy
- Client decides which third parties to trust

PKI – Public Key Infrastructure

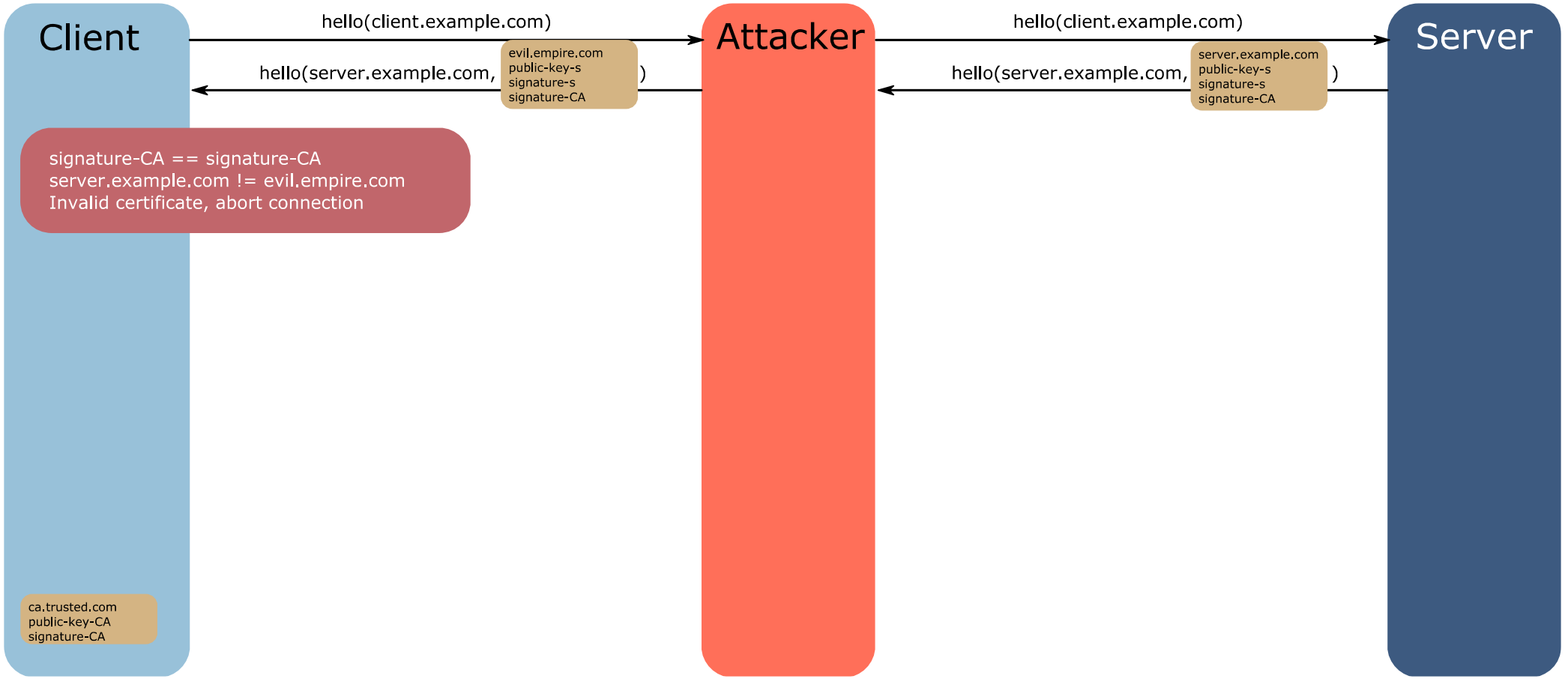
- You can sign a cryptographic signature
- A **Certificate Authority (CA)** is an entity that the client trusts which can sign certificates of servers
- A **Certificate Signing Request (CSR)** holds an incomplete certificate signed by the server's private key that needs to be signed by the CA
- The client only needs to hold the certificates of select trusted CAs and every server needs to have a certificate signed by a trusted CA.

ca technologies

What is Missing?



Host Name Validation



Host Name Validation

- Whenever someone asks a CA for a signature, the CA must validate that the requester owns the domain
- Every trusted CA must do this for every request
- A mistake by any CA the client trusts leads to MITM possibility

Host Name Validation – Types of Certificates

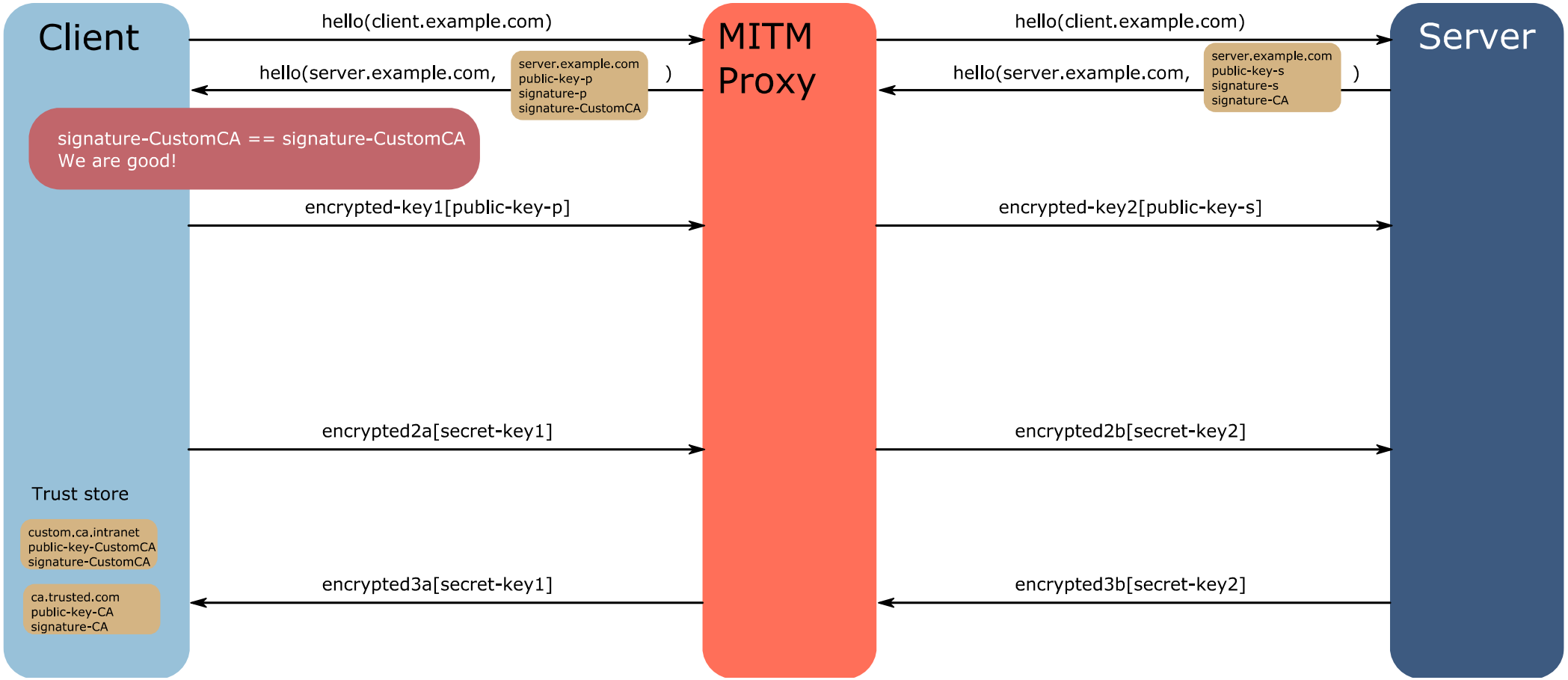
- Domain Validation (DV) – Owner of the domain responds to email in DNS record or a challenge is posted on the server hosted by that domain
- Organization Validation (OV) - Perform additional checks on top of DV by checking the organization exists and the applicant can request the certificate
- Extended Validation (EV) – Like OV, but more rigorous. EV can’t be issued to individuals and to certain types of organizations.
- EV is displayed differently in the browser, no easy way to tell OV from DV

ca technologies

Custom Certificate Authority

- Clients trusts all certificate authorities it has in its trust store.
- You can run your own CA as long as you deploy its certificate to all clients
- Use cases:
  - Quickly sign certificates on private network
  - Transparent TLS proxy – legitimate MITM attack (e.g. for compliance purposes)

Transparent TLS Proxy – Legitimate MITM



Custom CA – Security Implications

Suppose the attacker gets hold of the private key of your custom CA.

- Since clients trust all certificate authorities the same, the attacker could MITM connection to any HTTPS protected website.
- Some clients only connect to certain servers (e.g. back-end services talking to other back-end services on a private network)
- Other clients (browsers) connect to the wide internet.
- A custom CA installed in the browser can be used to MITM connections it was not meant to protect (connections to download software, connections to your bank…)

ca technologies

Self-signed Certificates

- A self signed certificate is signed by the server's private key rather than by a CA
- A self signed certificate stored in the client's trust store is secure as long as it has been delivered securely.
- Getting a browser warning and clicking "I understand the implications" is not secure delivery.
- Browsers keep making it harder to accept a connection with untrusted certificate.

When Things Go Wrong

- CA private key leaked
- CA made to issue a fraudulent certificate
- Server private key leaked

One of the Most Famous Cases

- On September 2011 the trusted root CA www.diginotar.nl was compromised and issued several fraudulent certificates for popular domains such as google.com
- The target of the subsequent MITM attack were users in Iran
- The fraudulent certificates were discovered and published
- By October 2011 the CA trust was removed by all browsers and the company went bankrupt.
- Google “diginotar black tulip” for the full report.

ca technologies

Certificate Revocation

- Certificate revocation lets a CA recall a wrongly issued certificate
- Google “Certificate revocation is broken”
- Alternatively “Certificate revocation is broken [add current year]”
- Several methods, CRL, OCSP, OCSP stapling…
- All have problems with availability => Browsers implement soft-fail mechanisms
- All can be bypassed by a MITM attacker holding a fraudulent certificate
- In practice, browsers distribute a subset of the full list of revoked certificates via their updates.

Certificate Transparency

- All certificates issued by a root CA are saved in a public log verifiable by independent parties
- Monitors – independent detectors of fraudulent certificates that watch the public logs
- Latest Chrome rejects certificates not found in CT logs.
- Only applies to root CAs (custom CAs are not required to log and are not checked by browsers)

ca technologies

DNS Certification Authority Authorization

- A special DNS record says which CAs can issue a certificate for a given domain.
- Mandatory for all root CAs since September 2017
- Protects against validation errors made by the CA
- Cannot protect against:
  - Compromised root CAs
  - Compromised custom CAs

HTTP Public Key Pinning

- Server says that its public key is valid for a certain period
- Upon first connect, client saves the key
- If the client encounters a different key, it rejects the connection

Who Do You Trust If You:

- Install a self signed certificate in the client
- Install a custom CA certificate in the client
- Install a client (where the CAs it trusts were chosen by the vendor)
- Install a CA signed certificate in the server

Did the presentation help you answer these questions?

Where To Go Next?

- This presentation only covered trust related problems
- There are other classes of attacks against TLS
- Luckily these can be mitigated by being up to date and scanning for common vulnerabilities, for example with https://testssl.sh/
- Book: Ivan Ristić: Bulletproof SSL and TLS

Tomáš Pavelka
Principal Architect
tomas.pavelka@ca.com