ETH zürich

On the Effective Prevention of TLS Man-In-The-Middle Attacks in Web Applications

Nikos Karapanos and Srdjan Čapkun, ETH Zurich

USENIX Security 2014

Server authentication is problematic

Server authentication is problematic

- Compromised CAs

Server authentication is problematic

- Compromised CAs

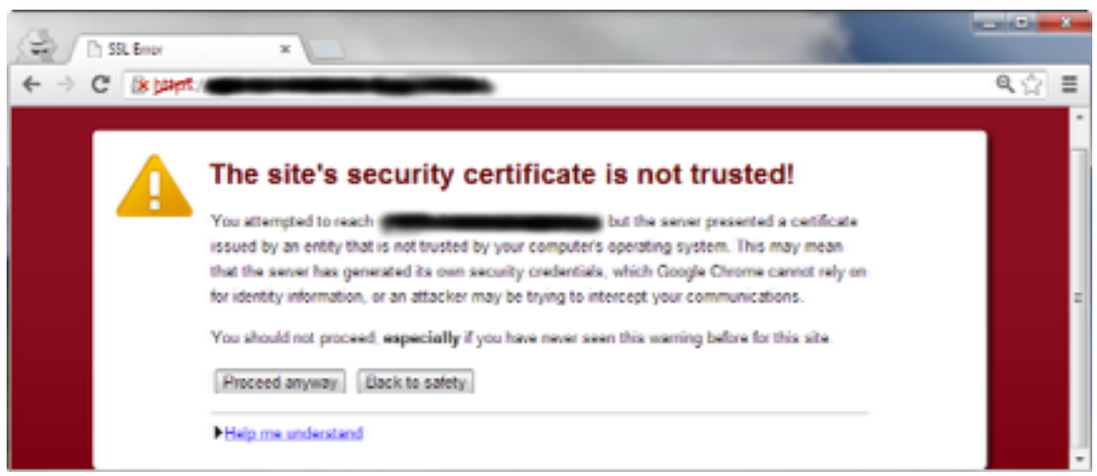
- Compromised server keys

Server authentication is problematic

- Compromised CAs

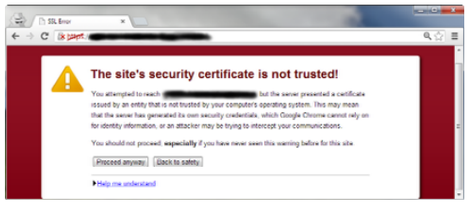
- Compromised server keys

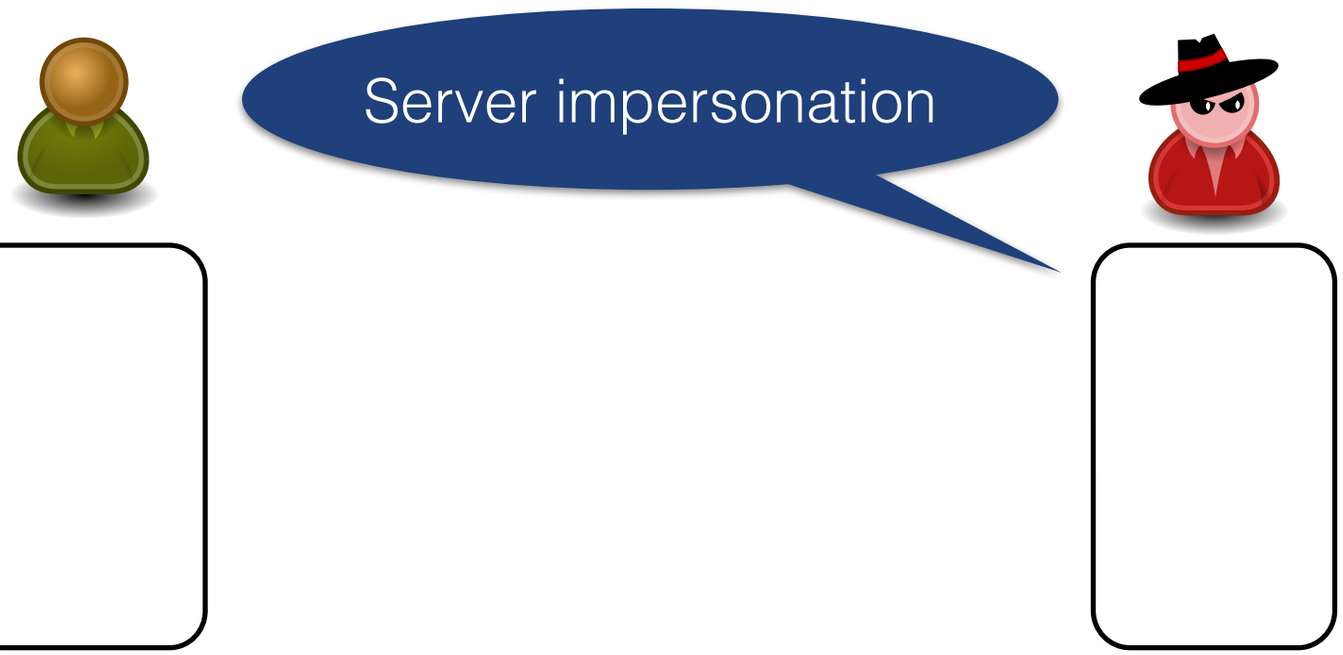
- Users click through warnings

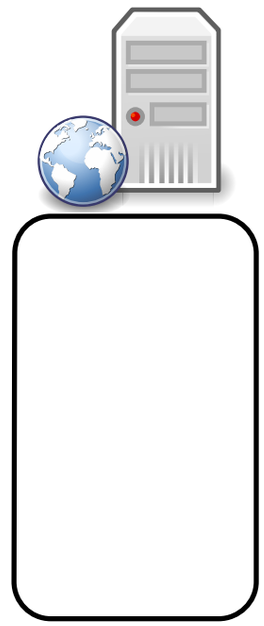


ETH zürich

Goal: Compromise user account

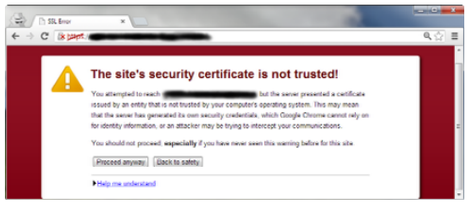






**Some of the icons used in this presentation were taken and adapted from opensecurityarchitecture.org*

Goal: Compromise user account



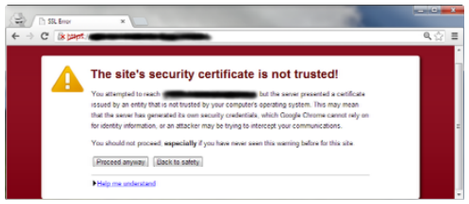
Server impersonation

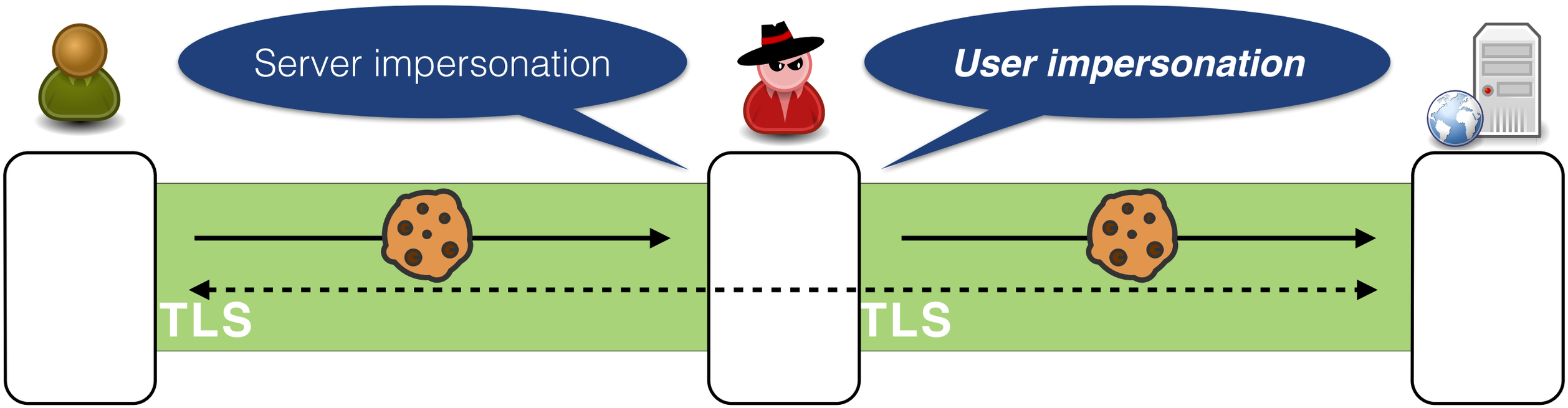
TLS

TLS

**Some of the icons used in this presentation were taken and adapted from opensecurityarchitecture.org*

Goal: Compromise user account

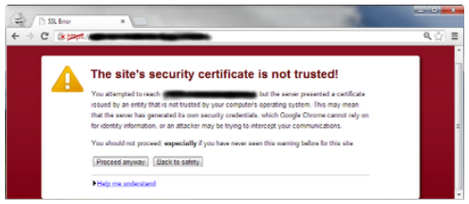


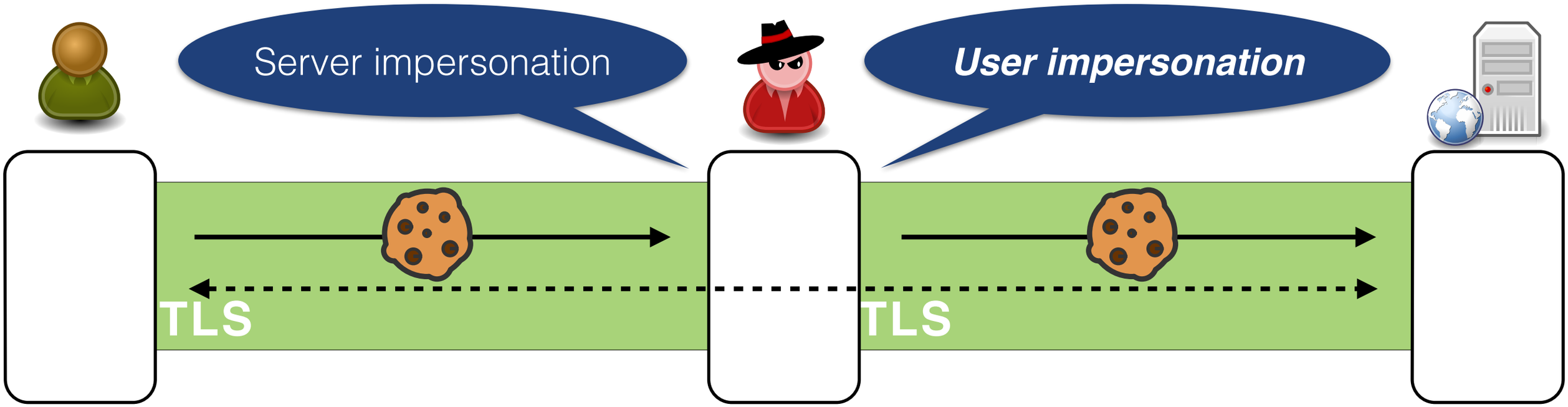


**Some of the icons used in this presentation were taken and adapted from opensecurityarchitecture.org*

ETH zürich

Goal: Compromise user account





TLS Channel IDs (Balfanz et al., IETF Internet Draft) proposed as a solution

**Some of the icons used in this presentation were taken and adapted from opensecurityarchitecture.org*

Main Contributions

Main Contributions

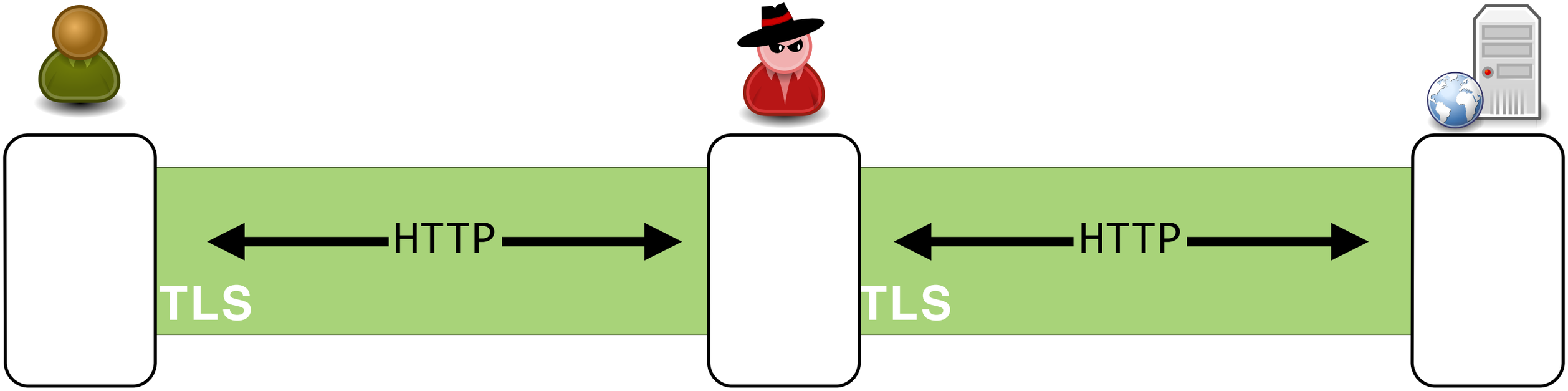
ETH zürich

1. We show an attack against TLS Channel IDs
   - extends usually considered attacker models
   - implemented and tested

Main Contributions

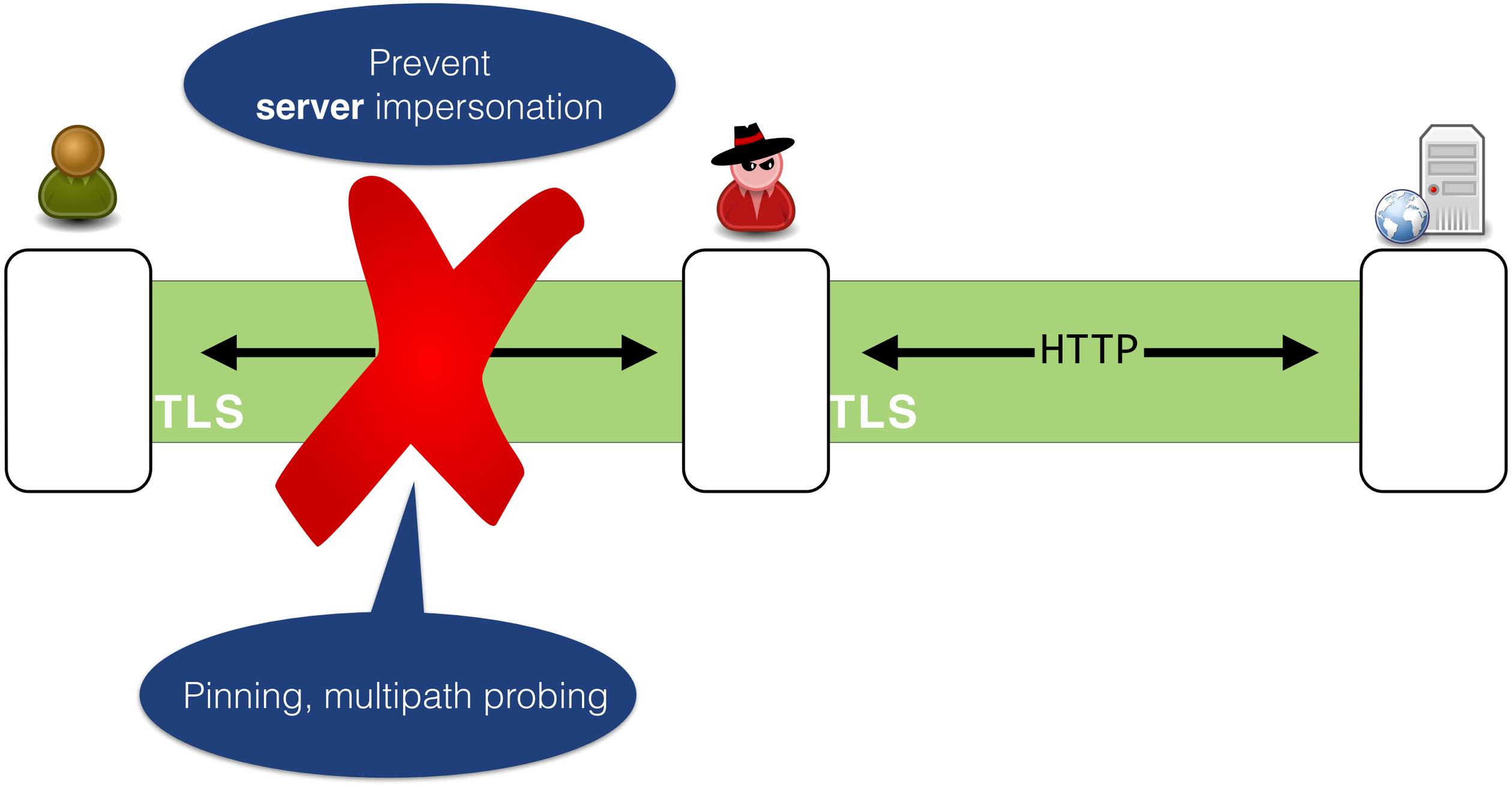
1. We show an attack against TLS Channel IDs
   - extends usually considered attacker models
   - implemented and tested
2. We propose a new solution: SISCA (Server Invariance with Strong Client Authentication)
   - prevents MITM attacks **even under server impersonation**
   - prototype implemented

Solutions focus on either endpoint

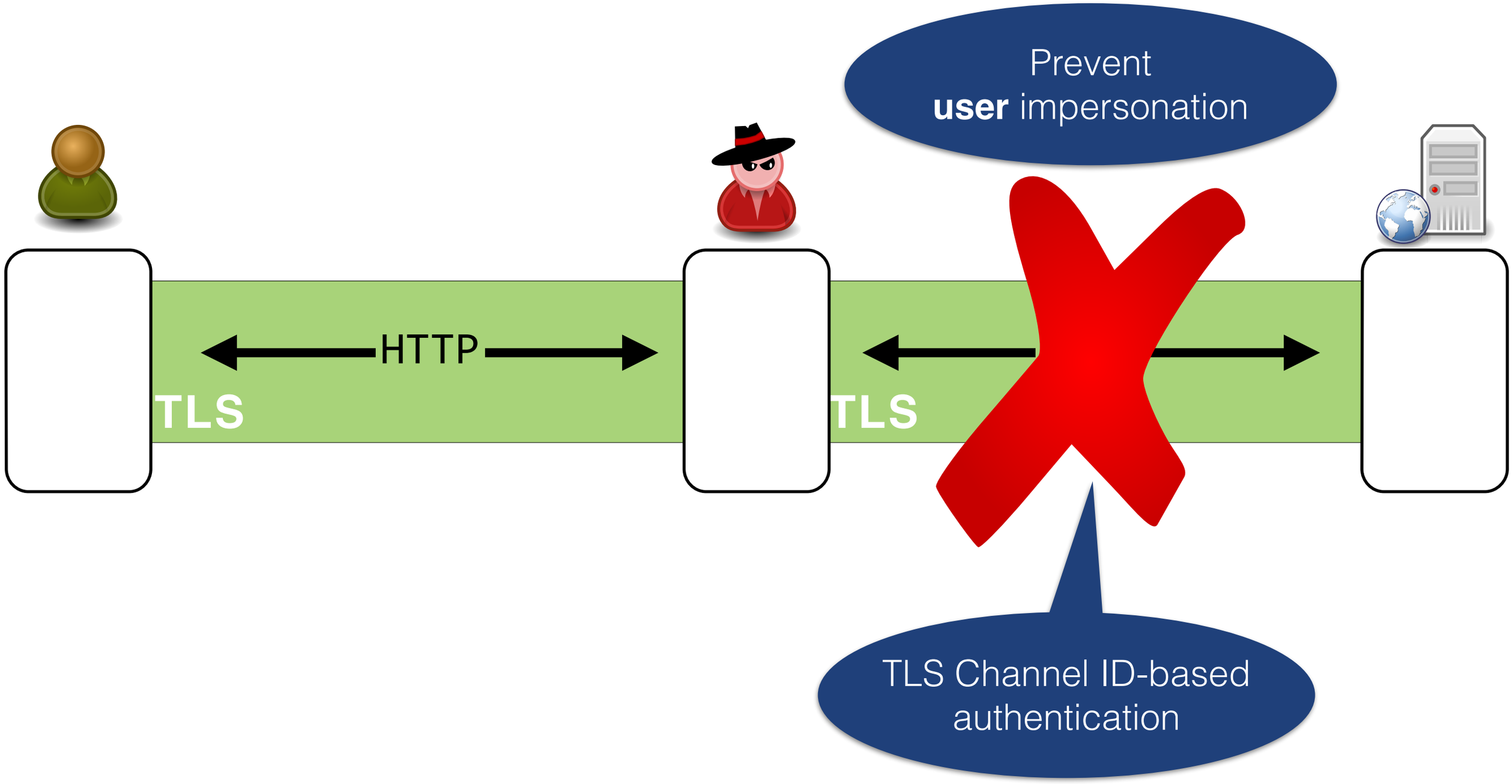


ETH zürich

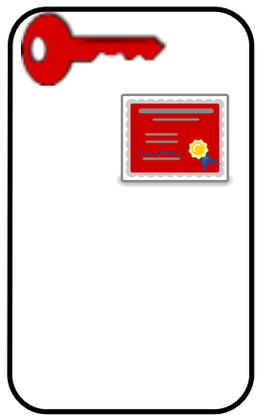
Solutions focus on either endpoint



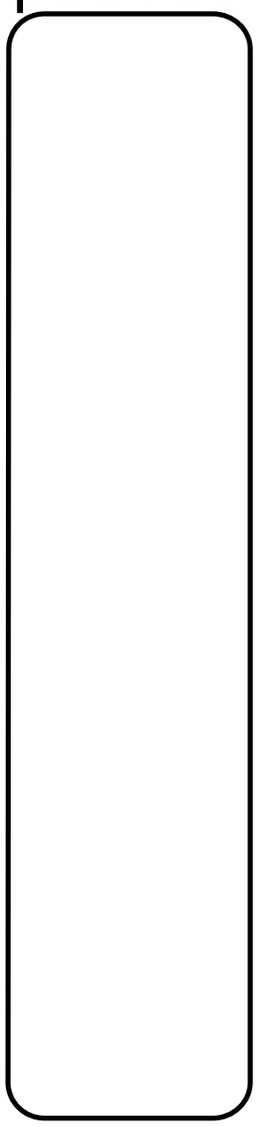
Solutions focus on either endpoint



Channel ID = public key of a private/public key pair



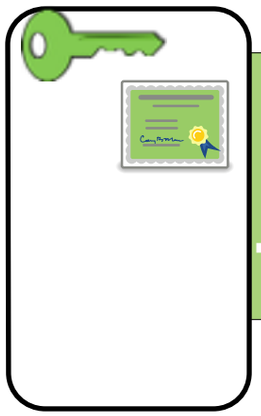
www.example.com



 ,  : **TLS Channel IDs**

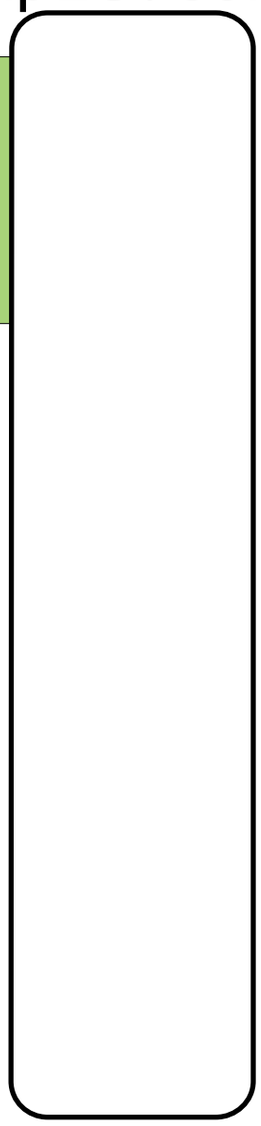
Channel ID = public key of a private/public key pair

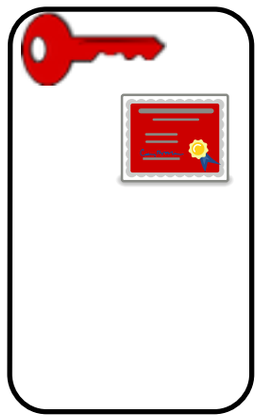
www.example.com



Here is my channel ID  , signed with the corresponding private key

TLS



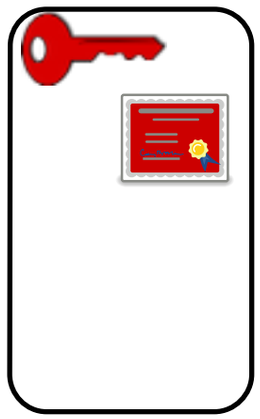


 ,  : TLS Channel IDs

Channel ID = public key of a private/public key pair

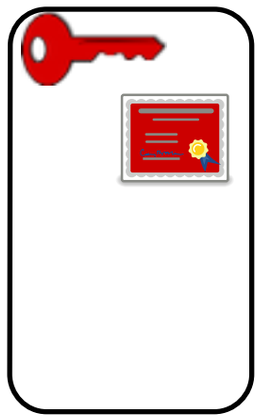
www.example.com

TLS

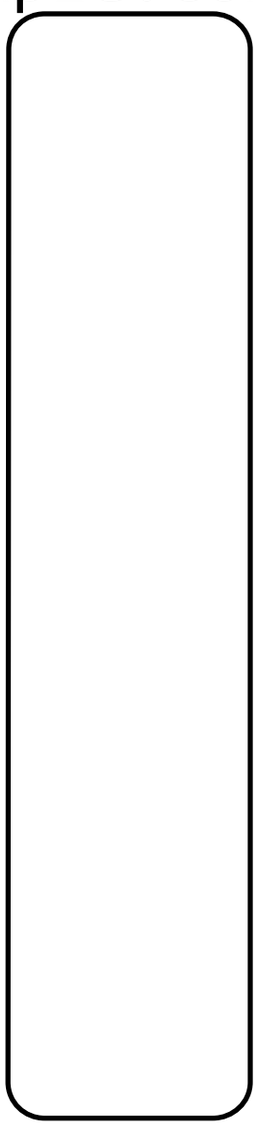


 ,  : TLS Channel IDs

Channel ID = public key of a private/public key pair



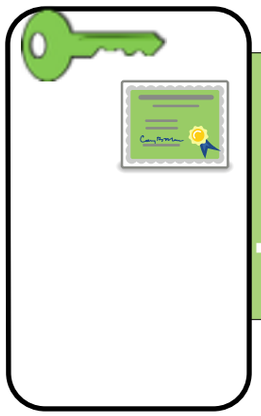
www.example.com



 ,  : **TLS Channel IDs**

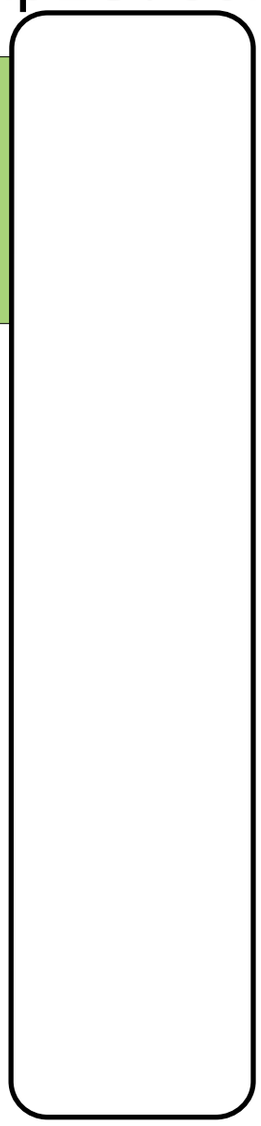
Channel ID = public key of a private/public key pair

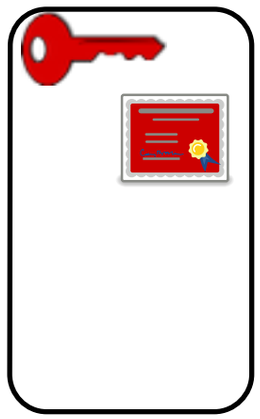
www.example.com



Here is my channel ID  , signed with the corresponding private key

TLS





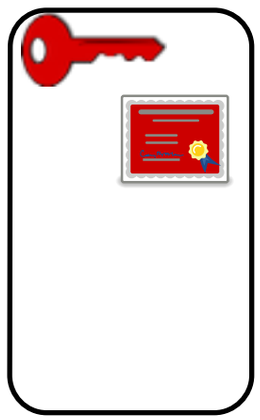
 ,  : TLS Channel IDs

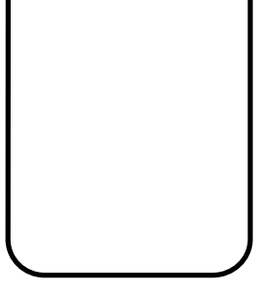
Channel ID = public key of a private/public key pair

www.example.com

TLS

A Channel ID identifies the same “TLS channel” across different TLS connections

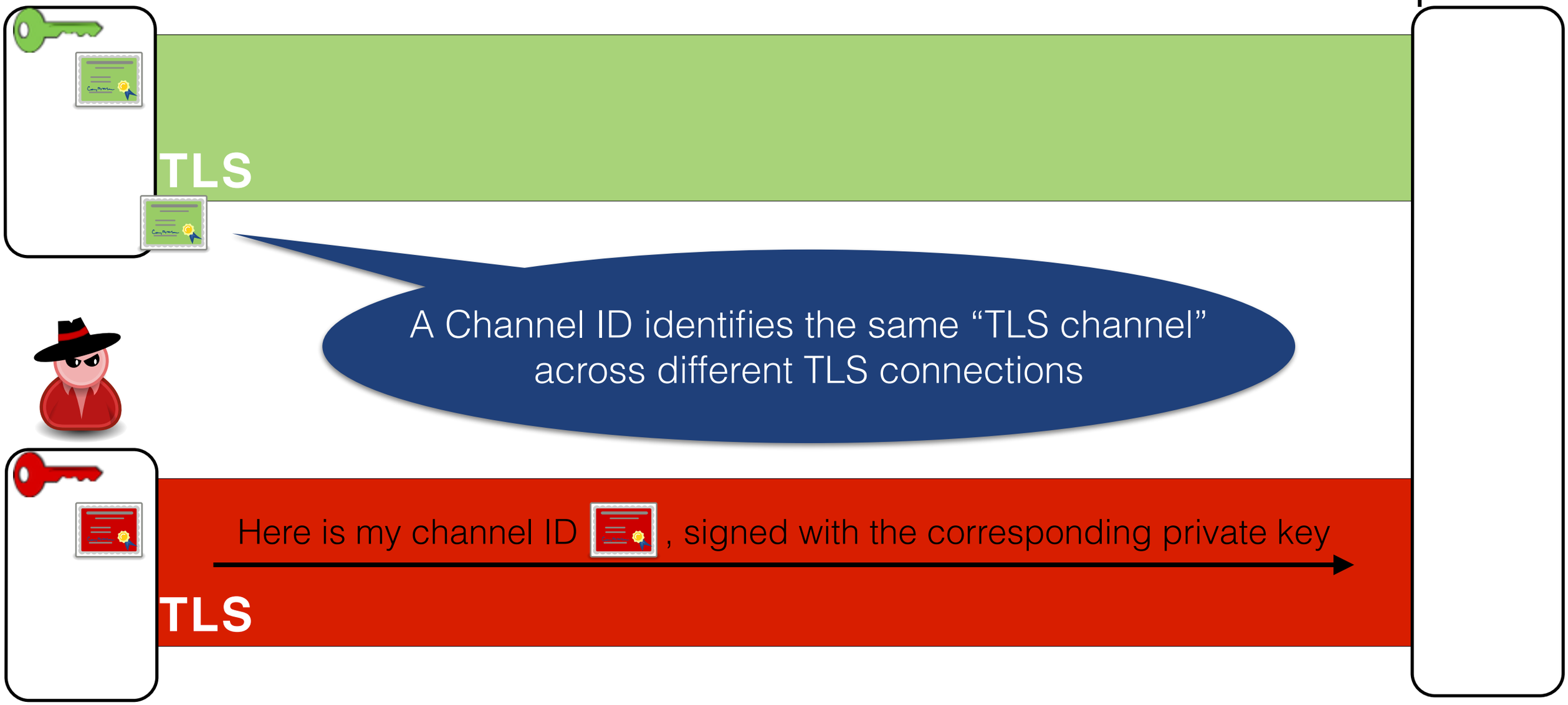


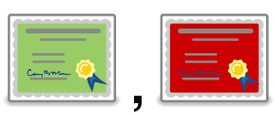


 ,  **: TLS Channel IDs**

Channel ID = public key of a private/public key pair

www.example.com





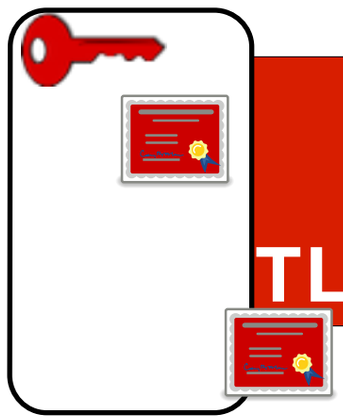
: TLS Channel IDs

Channel ID = public key of a private/public key pair

www.example.com

TLS

A Channel ID identifies the same “TLS channel” across different TLS connections



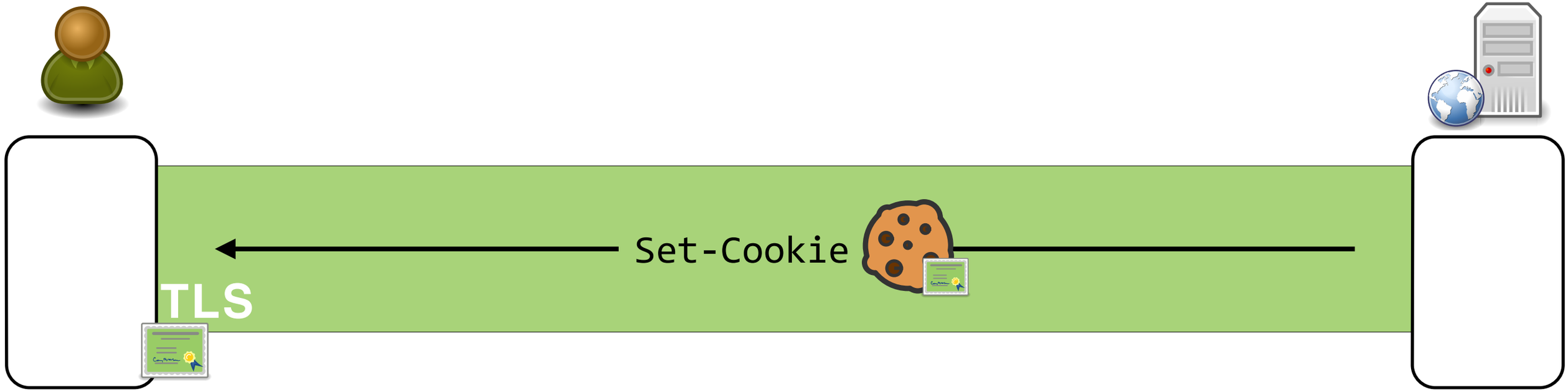
TLS

 ,  **: TLS Channel IDs**

Initial login (first login from a browser)

PhoneAuth (Czeskis et al., CCS 2012), FIDO Alliance U2F draft spec.

After initial login

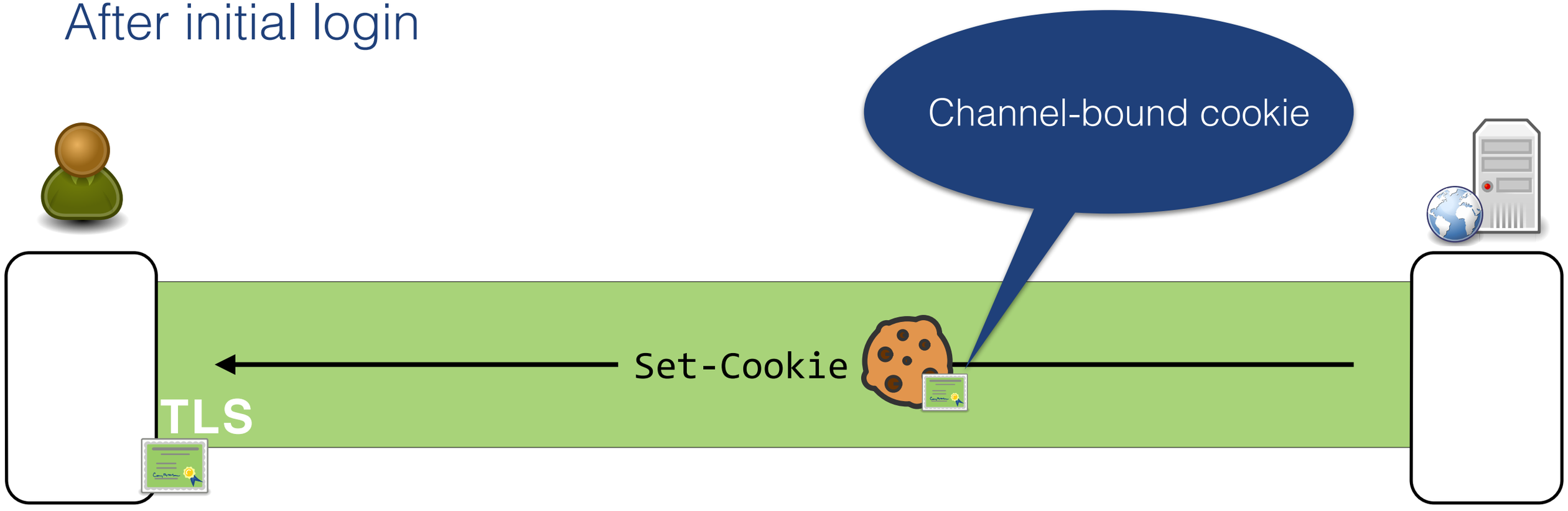


, : TLS Channel IDs

Initial login (first login from a browser)

PhoneAuth (Czeskis et al., CCS 2012), FIDO Alliance U2F draft spec.

After initial login



, : **TLS Channel IDs**

Initial login (first login from a browser)

PhoneAuth (Czeskis et al., CCS 2012), FIDO Alliance U2F draft spec.

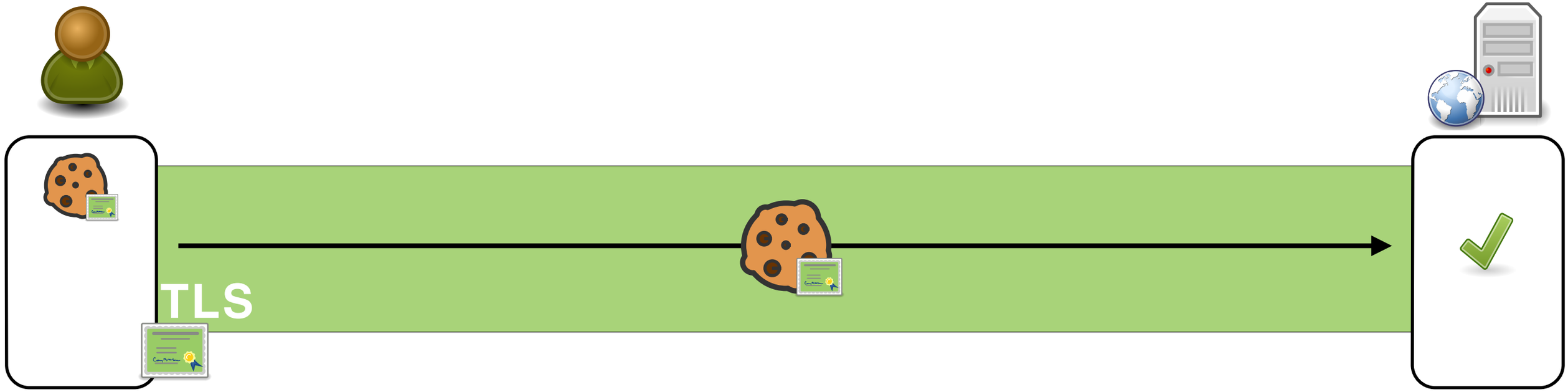
After initial login

, : TLS Channel IDs

Initial login (first login from a browser)

PhoneAuth (Czeskis et al., CCS 2012), FIDO Alliance U2F draft spec.

After initial login

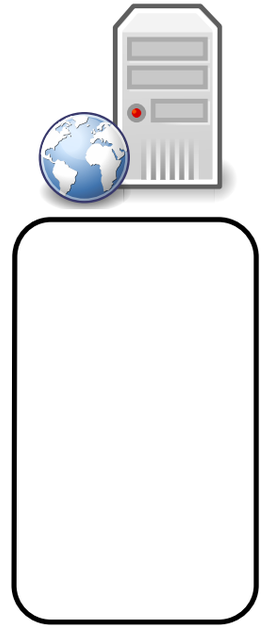


 , **: TLS Channel IDs**

Initial login (first login from a browser)

PhoneAuth (Czeskis et al., CCS 2012), FIDO Alliance U2F draft spec.

After initial login

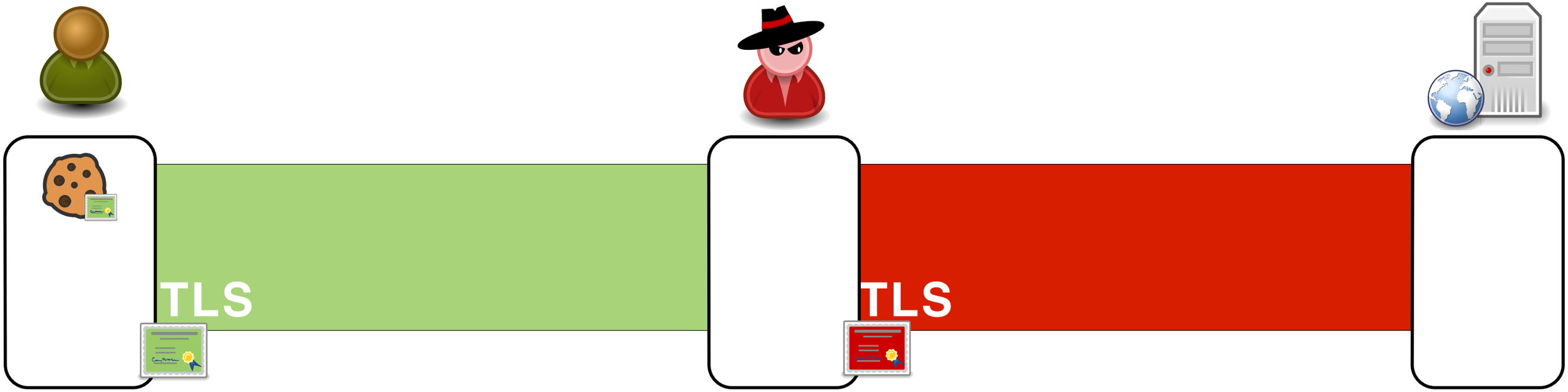


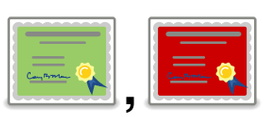
 , **: TLS Channel IDs**

Initial login (first login from a browser)

PhoneAuth (Czeskis et al., CCS 2012), FIDO Alliance U2F draft spec.

After initial login

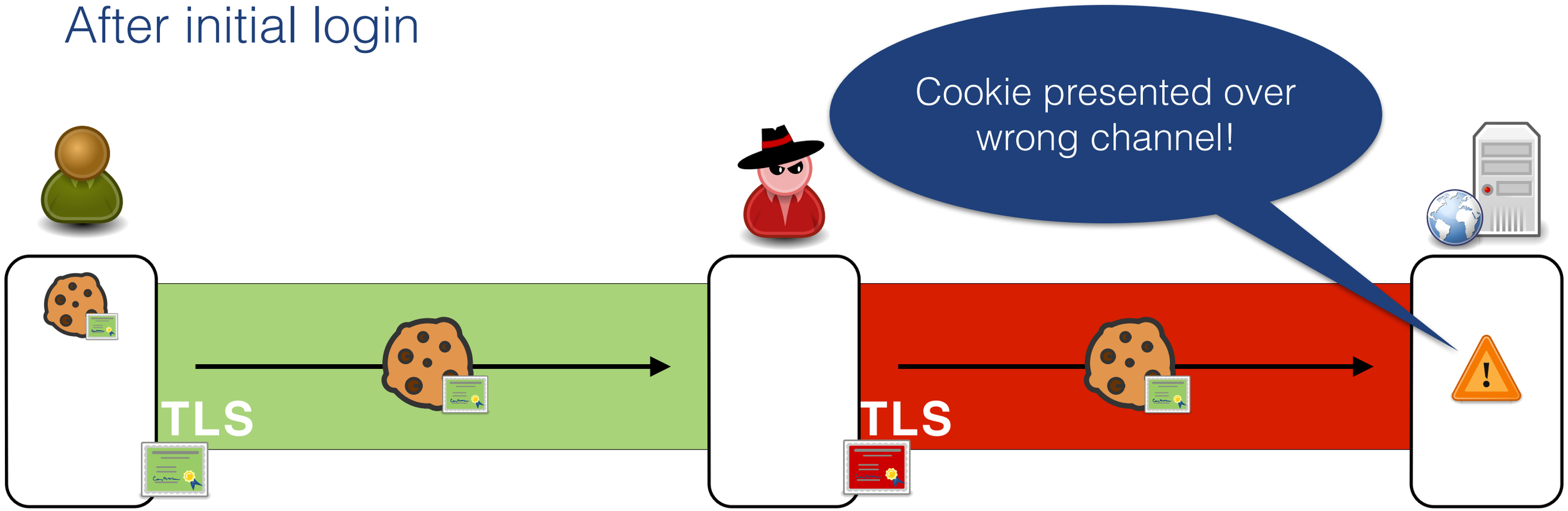


 : TLS Channel IDs

Initial login (first login from a browser)

PhoneAuth (Czeskis et al., CCS 2012), FIDO Alliance U2F draft spec.

After initial login

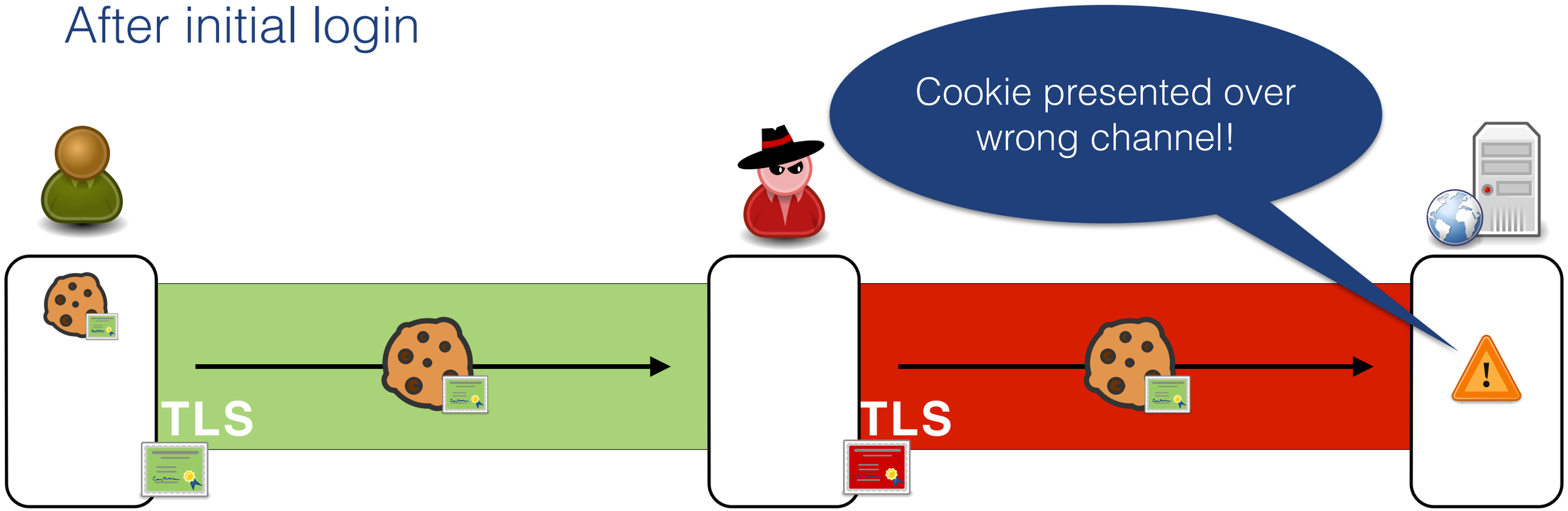


, : **TLS Channel IDs**

Initial login (first login from a browser)

PhoneAuth (Czeskis et al., CCS 2012), FIDO Alliance U2F draft spec.

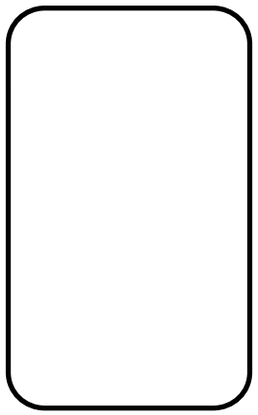
After initial login



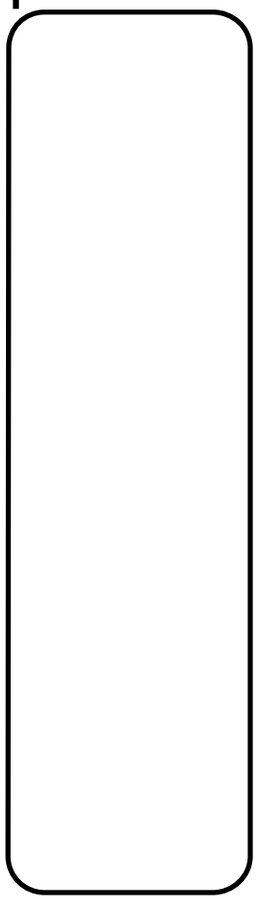
, : **TLS Channel IDs**

MITM-Script-In-The-Browser (MITM-SITB)

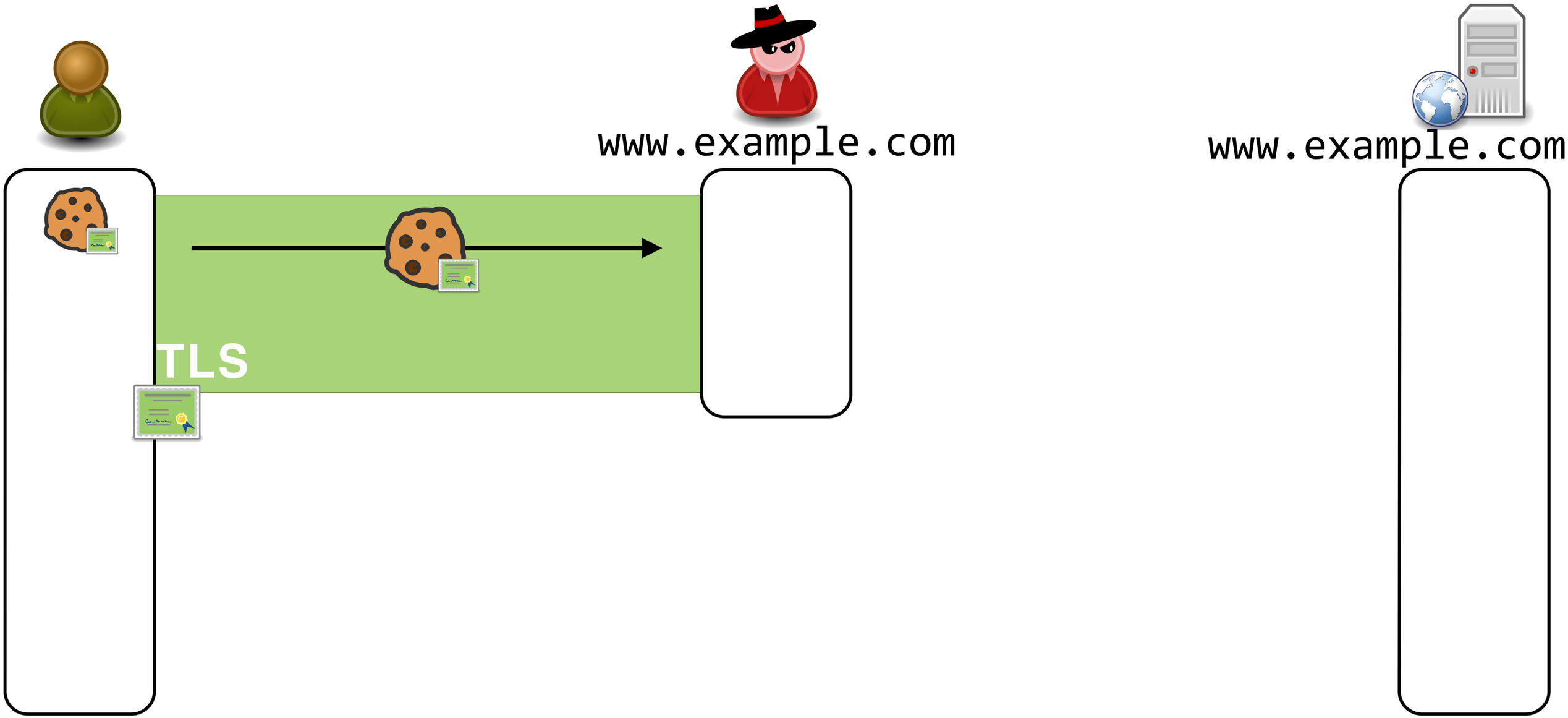
www.example.com



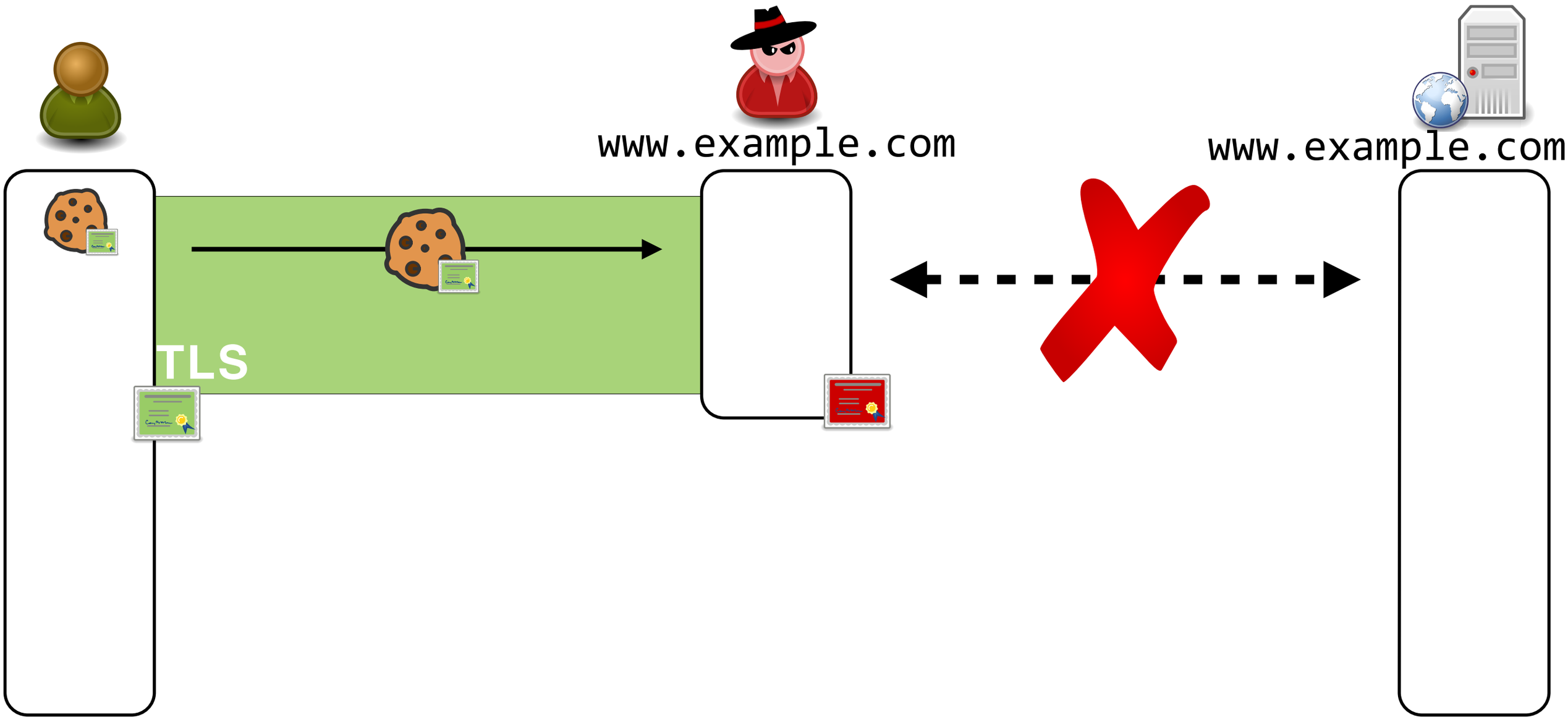
www.example.com



MITM-Script-In-The-Browser (MITM-SITB)

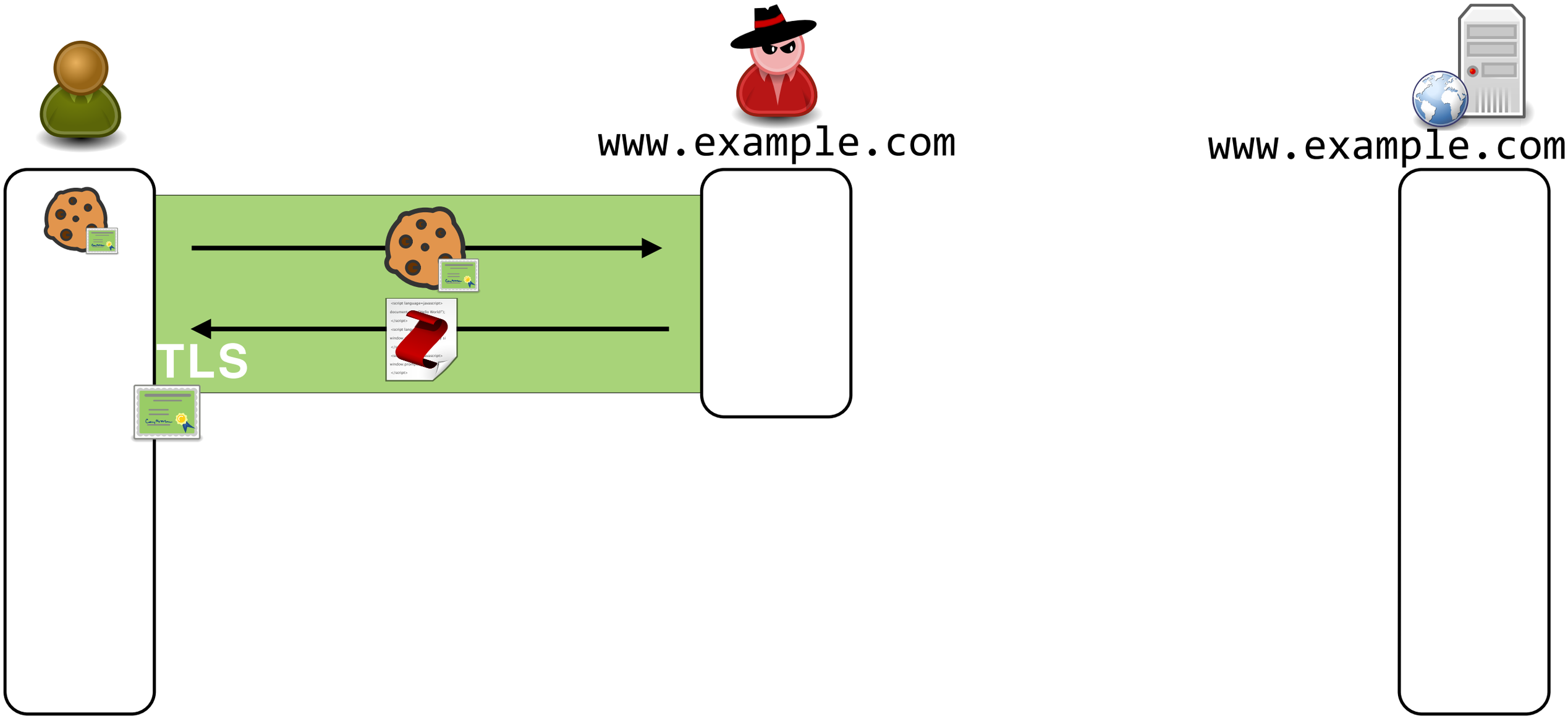


MITM-Script-In-The-Browser (MITM-SITB)

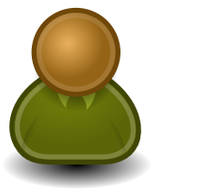


ETH zürich

MITM-Script-In-The-Browser (MITM-SITB)



MITM-Script-In-The-Browser (MITM-SITB)



www.example.com

www.example.com

Executed within the target server's web origin

MITM-Script-In-The-Browser (MITM-SITB)

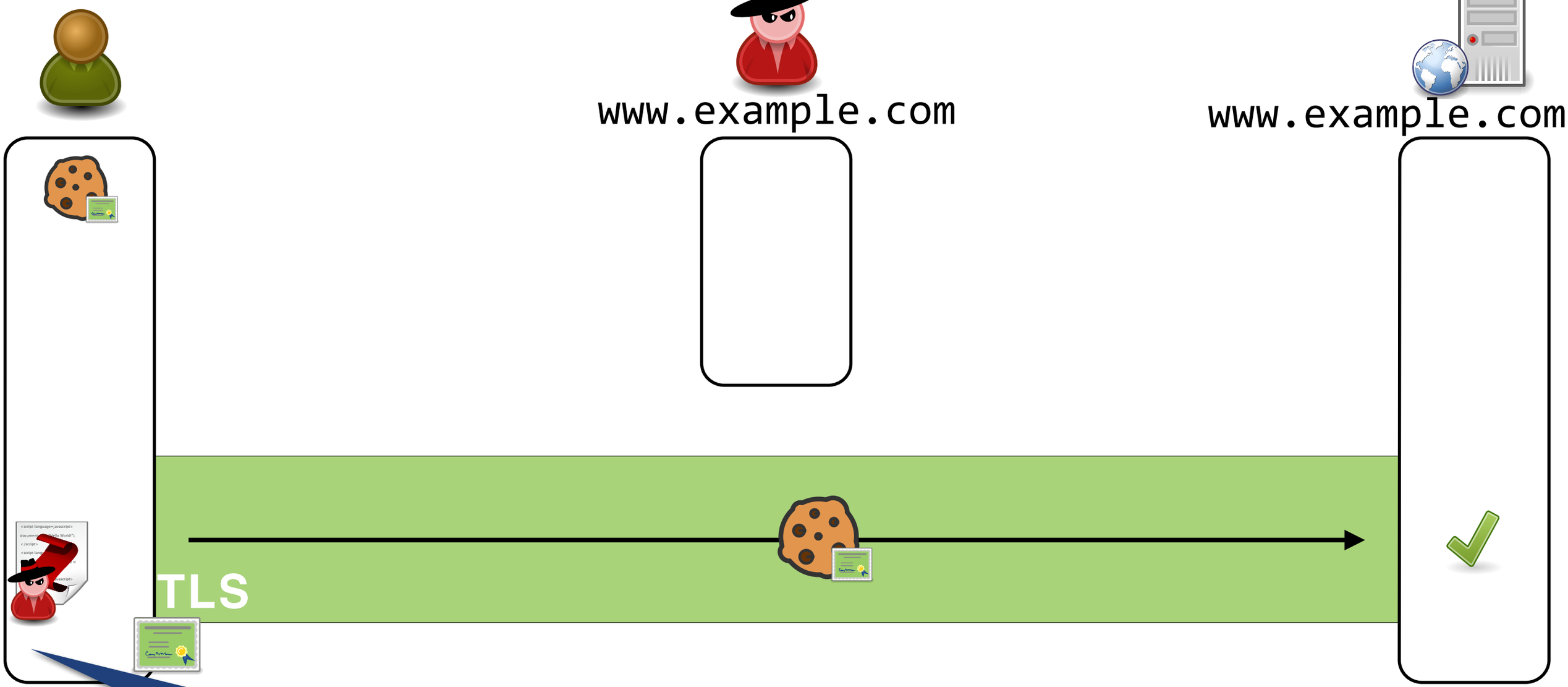
www.example.com

www.example.com

TLS

Executed within the target server's web origin

MITM-Script-In-The-Browser (MITM-SITB)



Executed within the target server's web origin

MITM-SITB was missed by a number of proposals

MITM-SITB was missed by a number of proposals

- TLS Channel IDs (PhoneAuth, FIDO U2F)
- TLS client auth., SSL/TLS session-aware user auth. (Oppliger et al, Computer Communications 2006)

MITM-SITB was missed by a number of proposals

- TLS Channel IDs (PhoneAuth, FIDO U2F)
- TLS client auth., SSL/TLS session-aware user auth. (Oppliger et al, Computer Communications 2006)
- These solutions focus on client authentication but ignore server authentication.

▸ *Attacker impersonates the server and injects malicious but "trusted" client-side code*

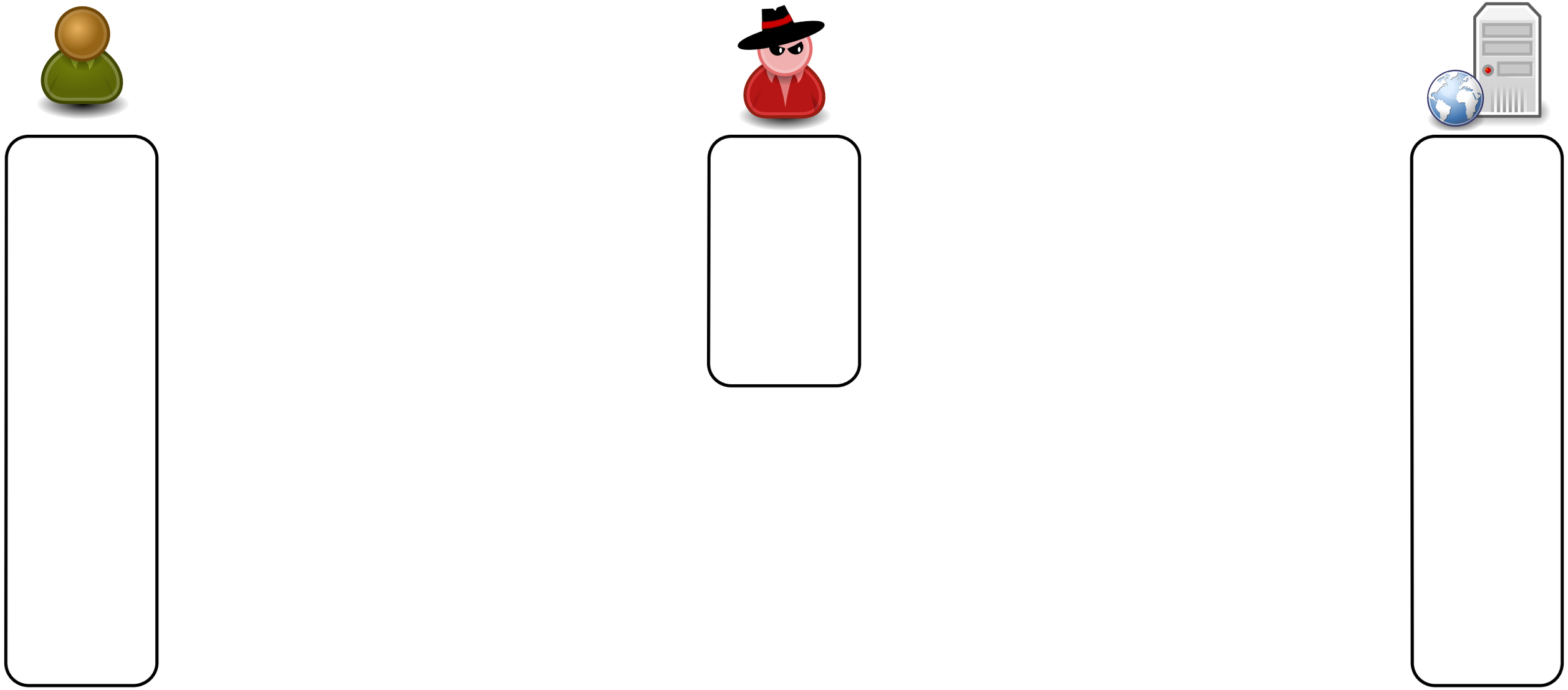
MITM-SITB was missed by a number of proposals

- TLS Channel IDs (PhoneAuth, FIDO U2F)
- TLS client auth., SSL/TLS session-aware user auth. (Oppliger et al, Computer Communications 2006)
- These solutions focus on client authentication but ignore server authentication.

- *Attacker impersonates the server and injects malicious but "trusted" client-side code*

=> we cannot ignore **server authentication**
But…

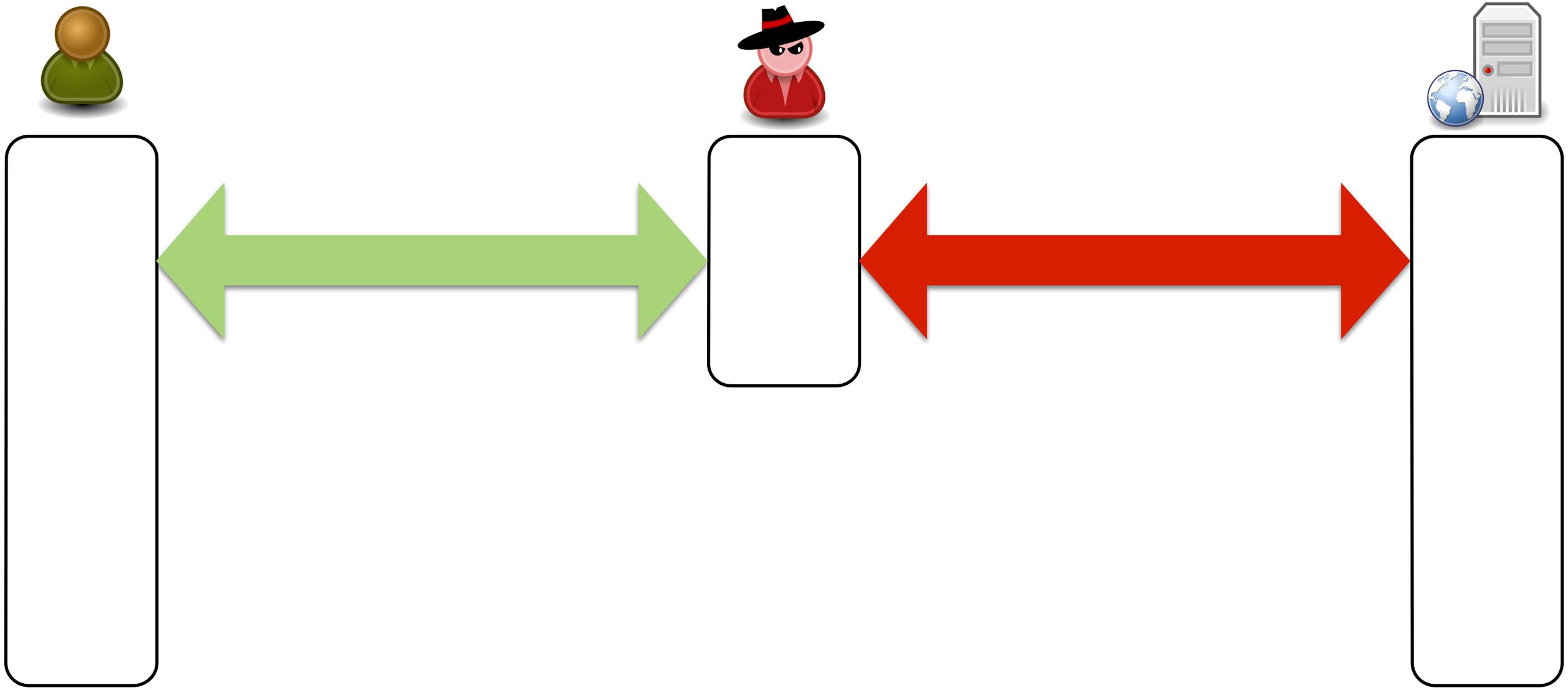
Insight



Do we really need server authentication?

ETH zürich

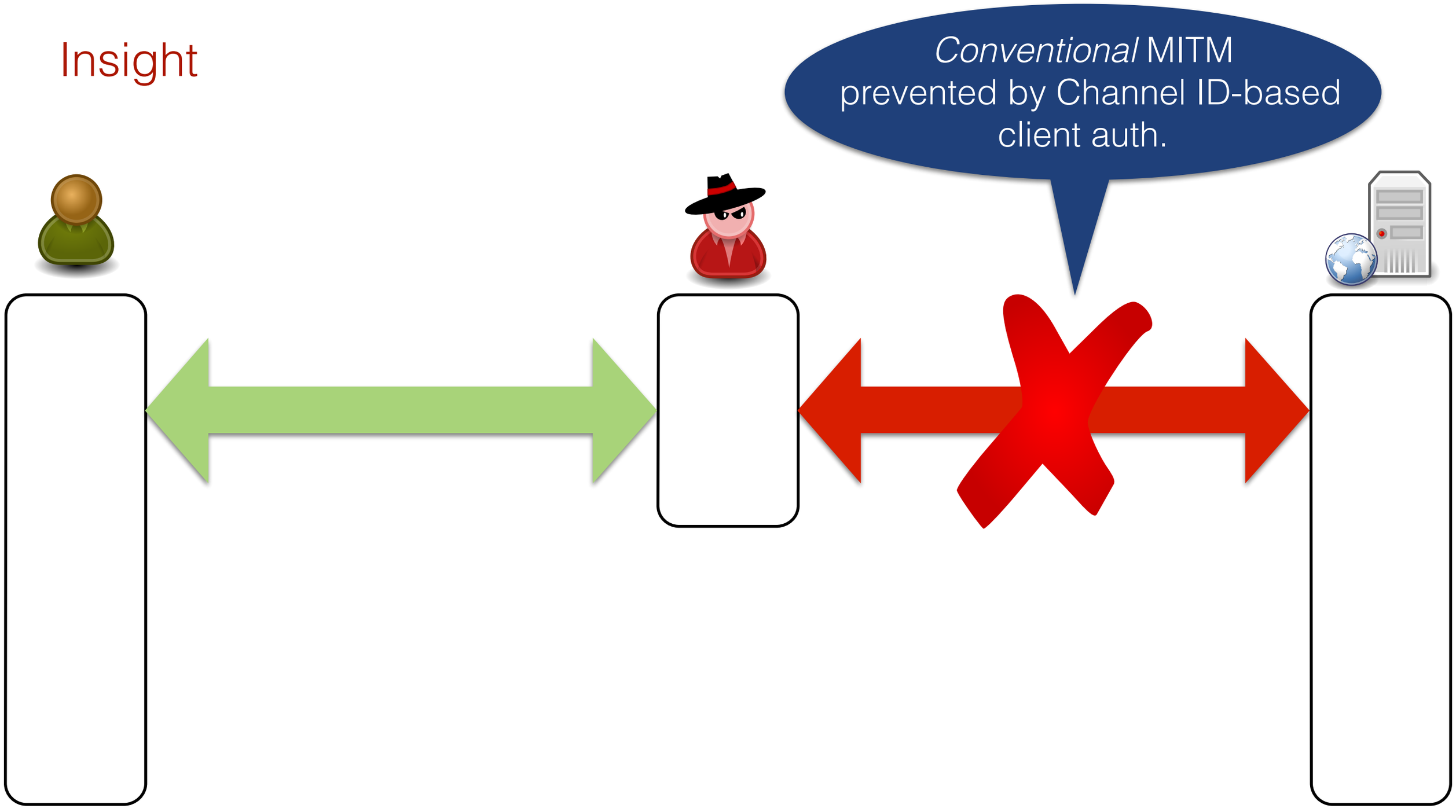
Insight



Do we really need server authentication?

ETH zürich

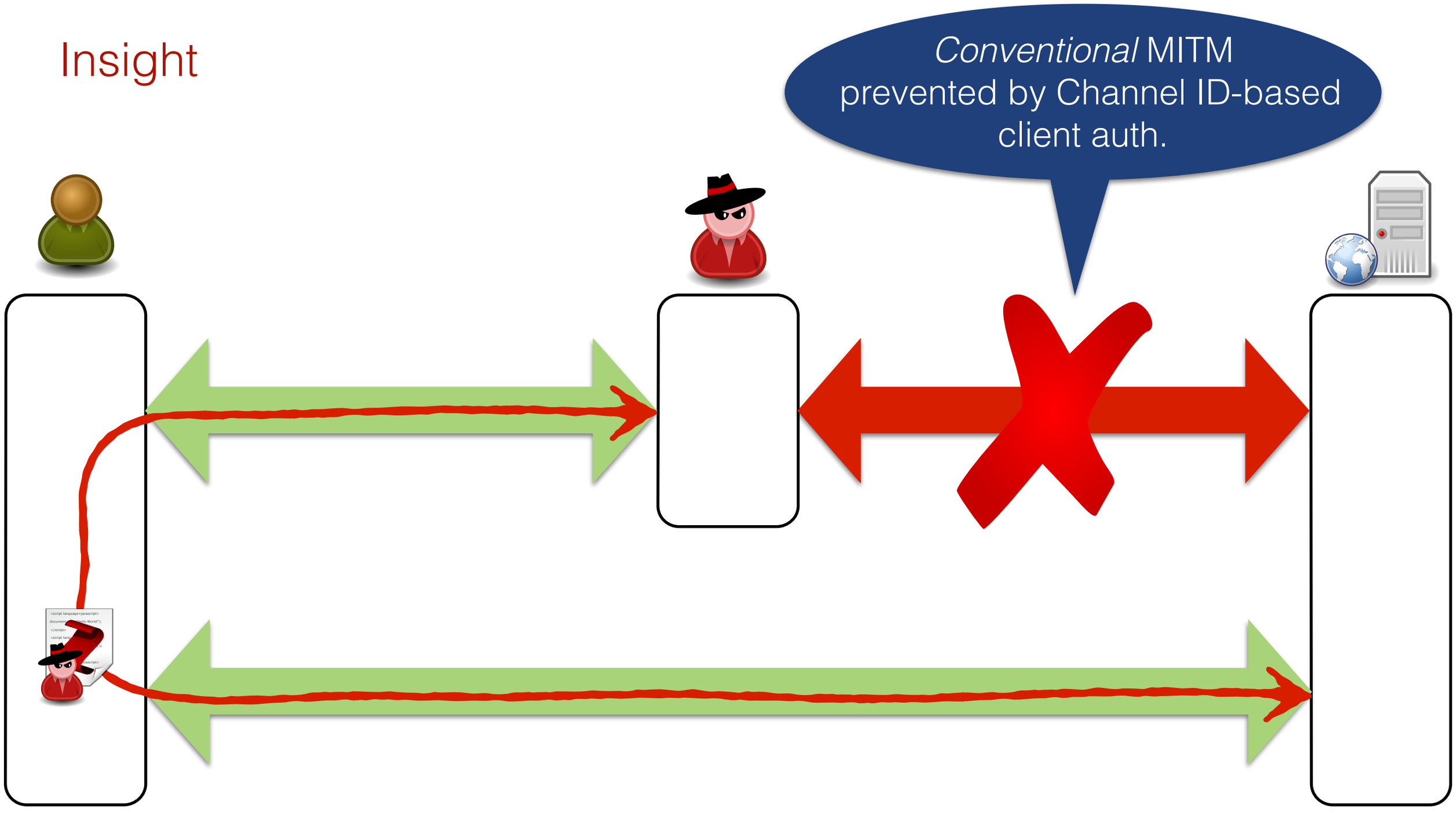
Insight



Do we really need server authentication?

ETH zürich

Insight

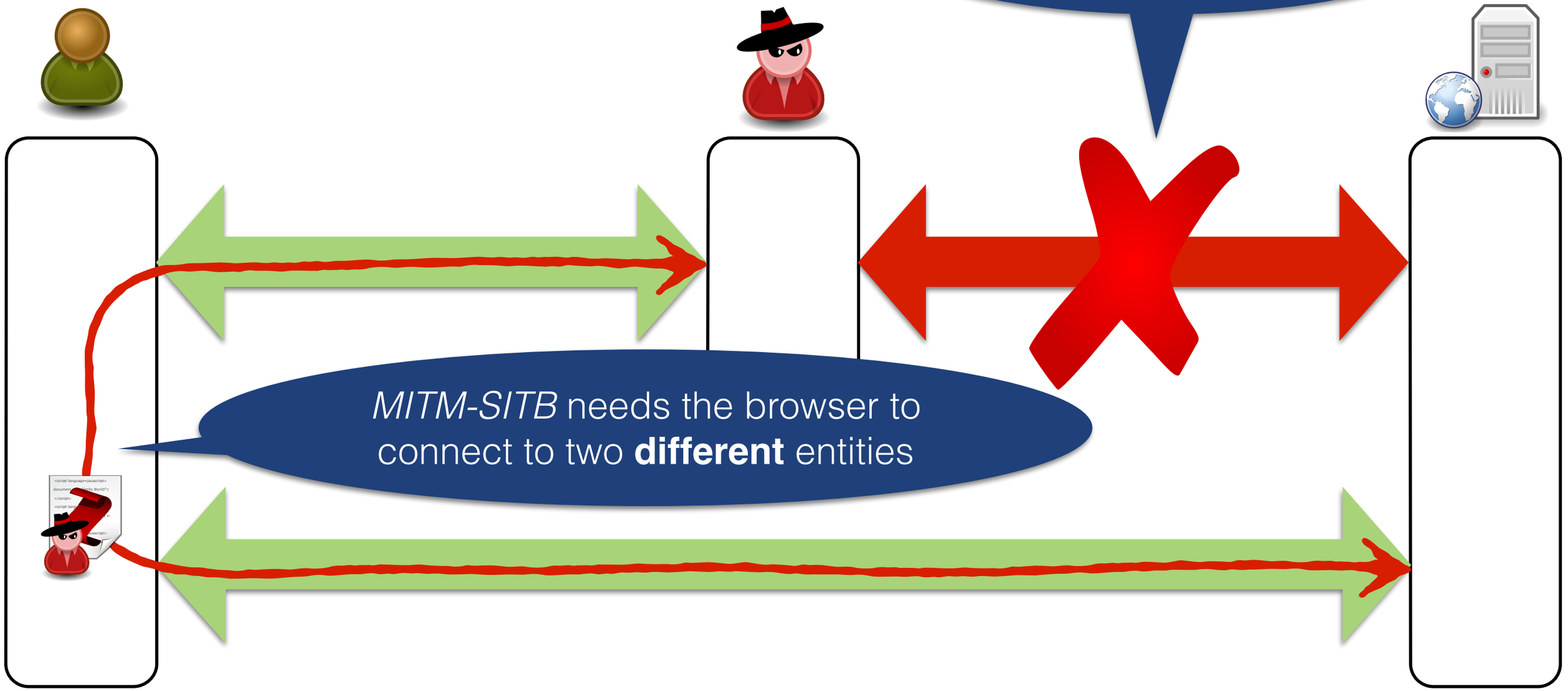


Do we really need server authentication?

ETH zürich

Insight

Conventional MITM prevented by Channel ID-based client auth.



MITM-SITB needs the browser to connect to two **different** entities

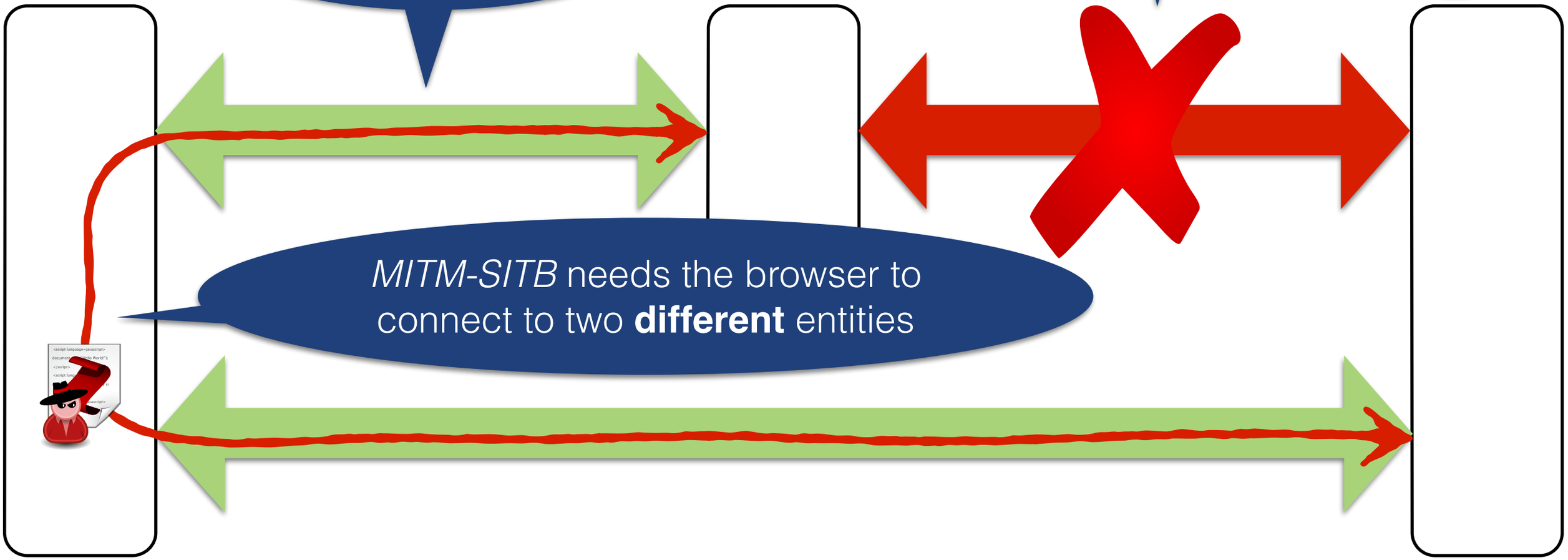
Do we really need server authentication?

ETH zürich

Insight

Conventional MITM prevented by Channel ID-based client auth.

1. Attacker server (inject code)



MITM-SITB needs the browser to connect to two **different** entities

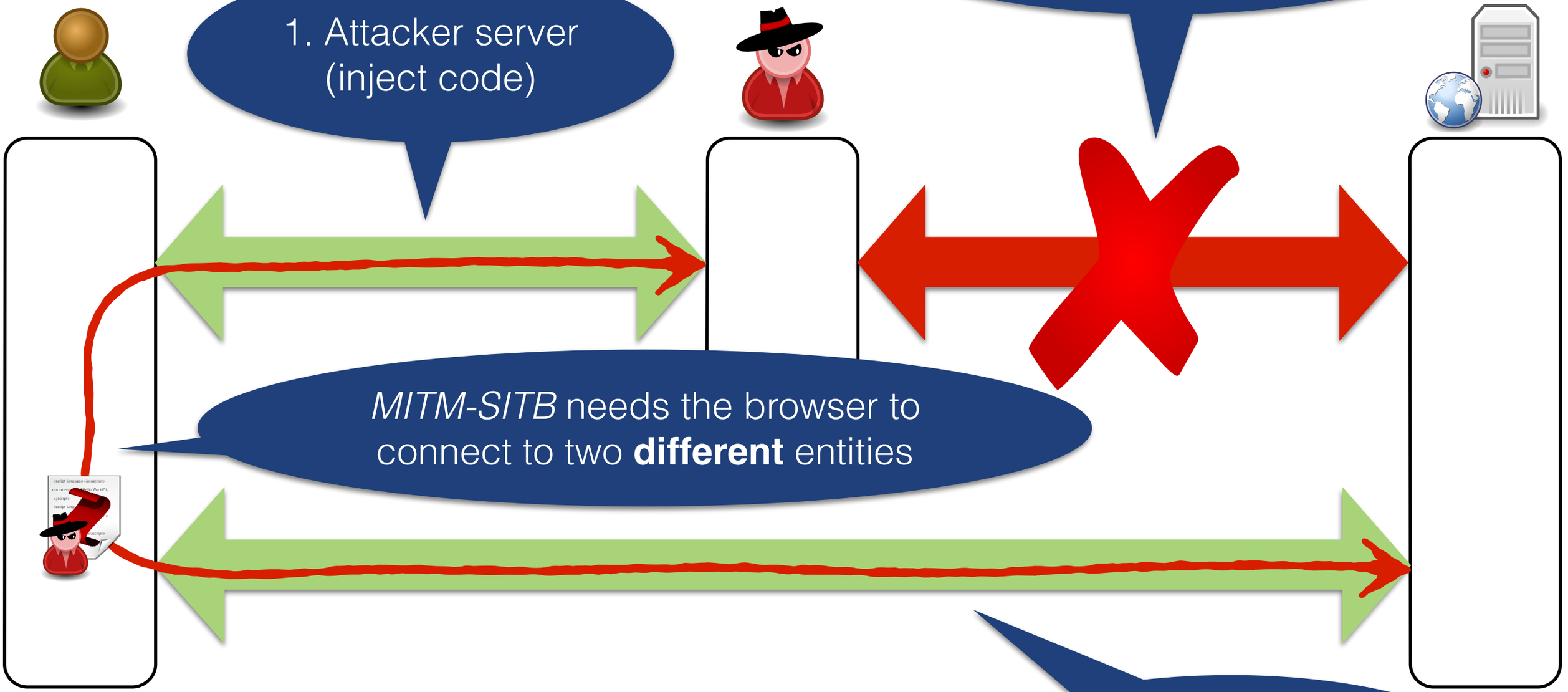
Do we really need server authentication?

ETH zürich

Insight

Conventional MITM prevented by Channel ID-based client auth.

1. Attacker server (inject code)



MITM-SITB needs the browser to connect to two **different** entities

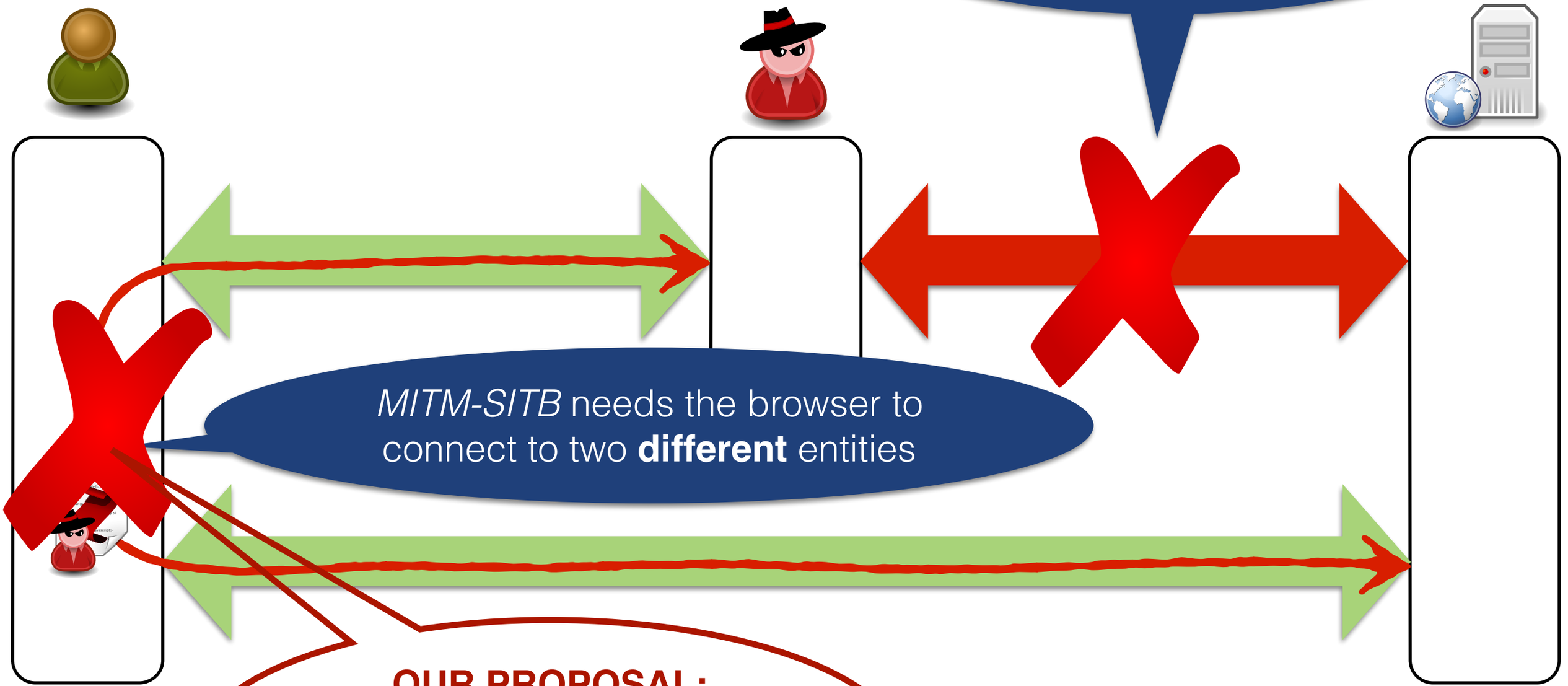
2. Legitimate server (access user account)

Do we really need server authentication?

ETH zürich

Insight

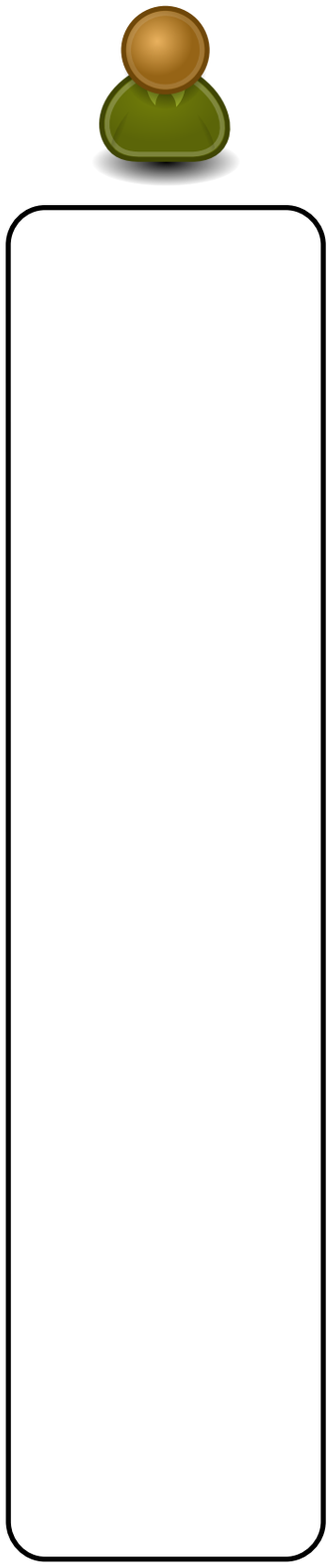
Conventional MITM prevented by Channel ID-based client auth.

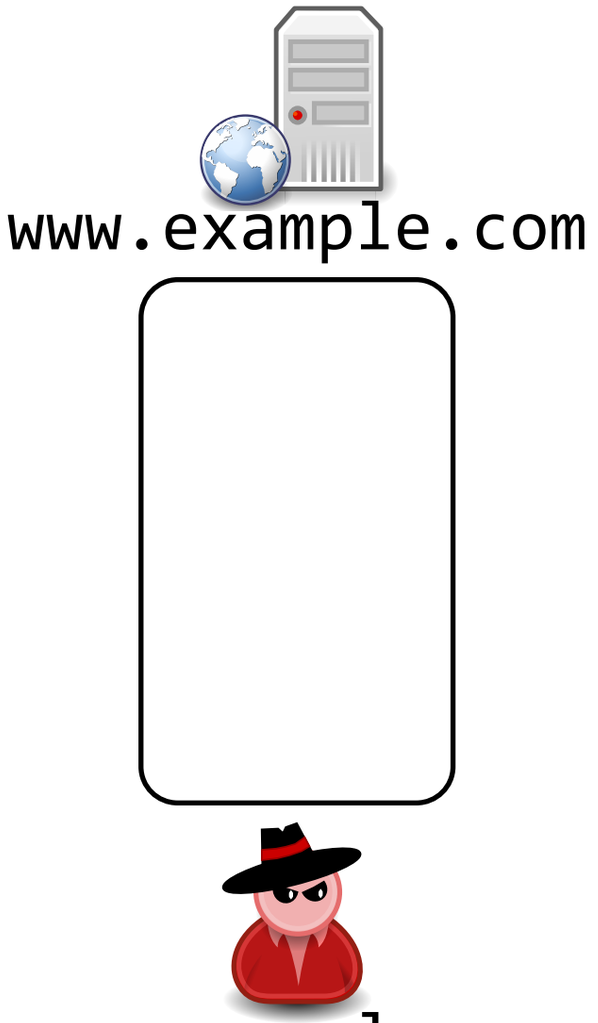


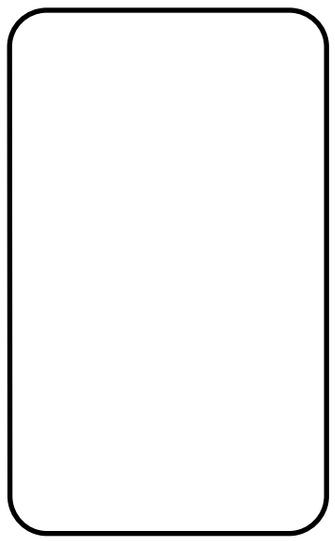
MITM-SITB needs the browser to connect to two **different** entities

OUR PROPOSAL:
ensure that the browser does not connect to different entities!

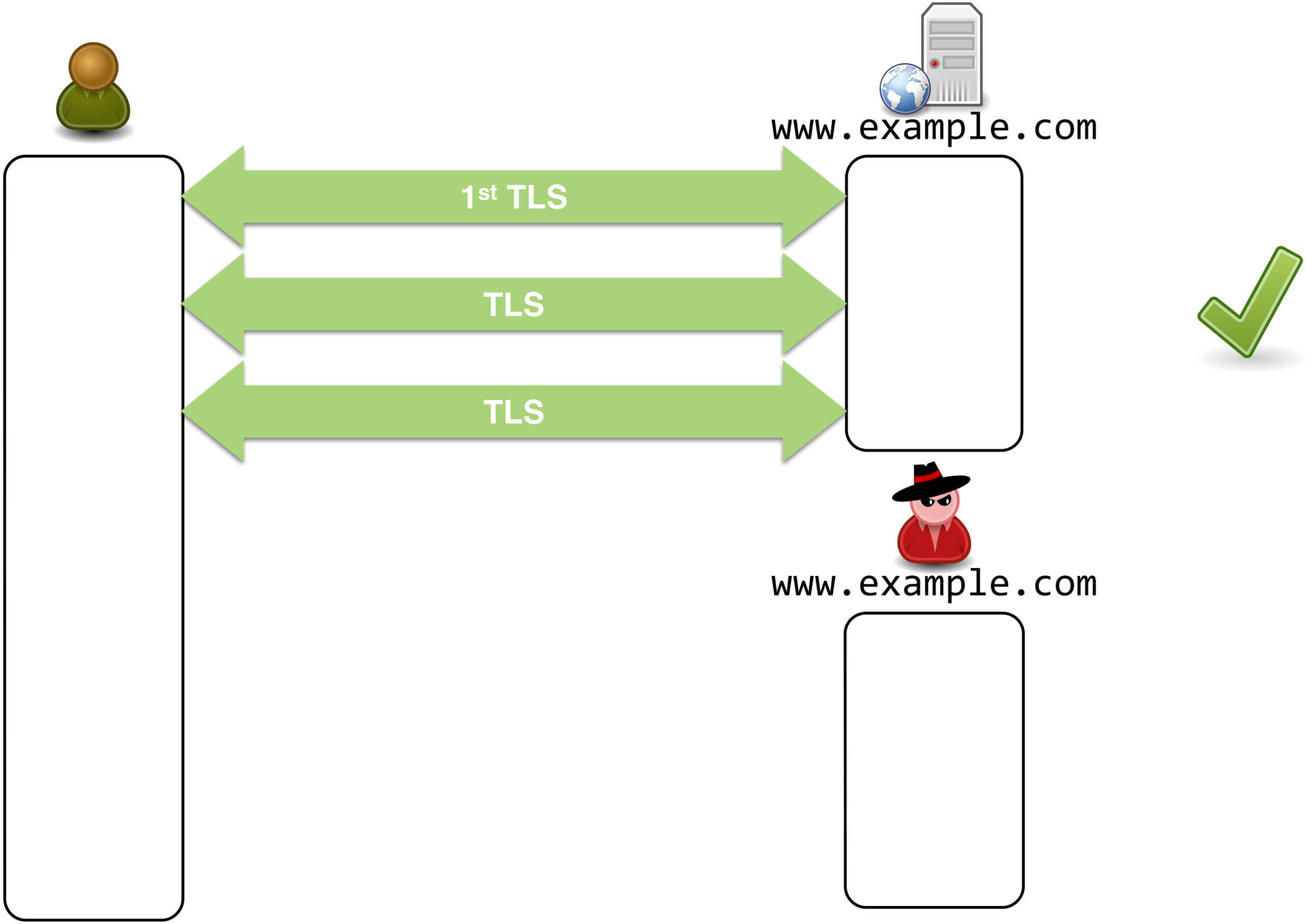
This Invariance is Enforced by the Browser



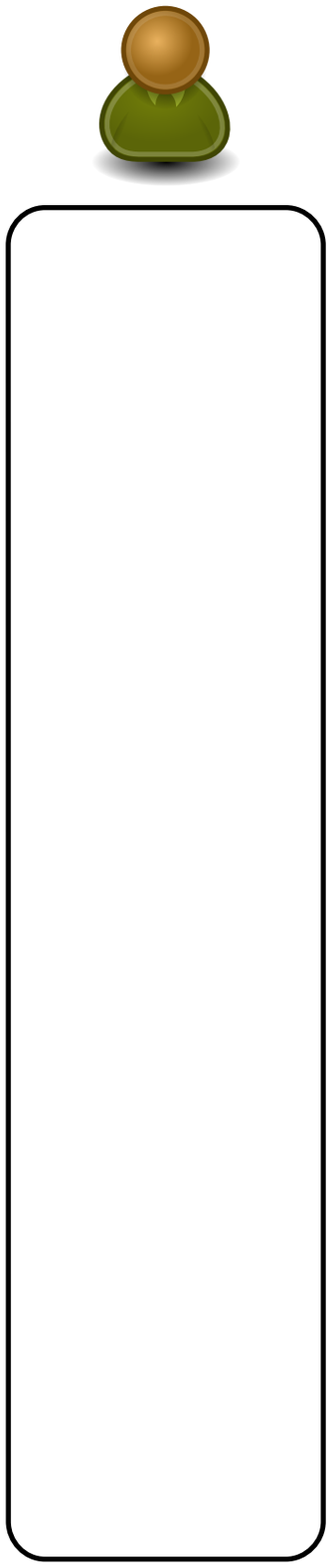


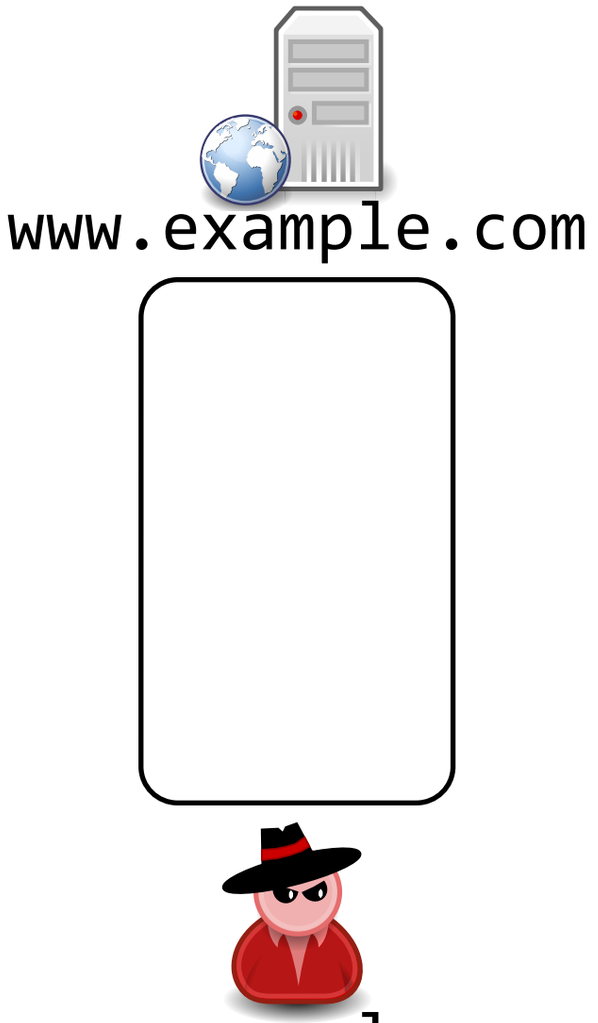


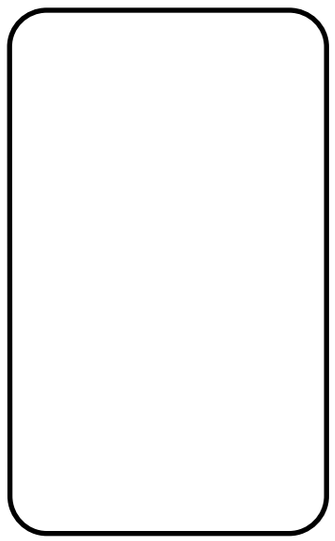
This Invariance is Enforced by the Browser



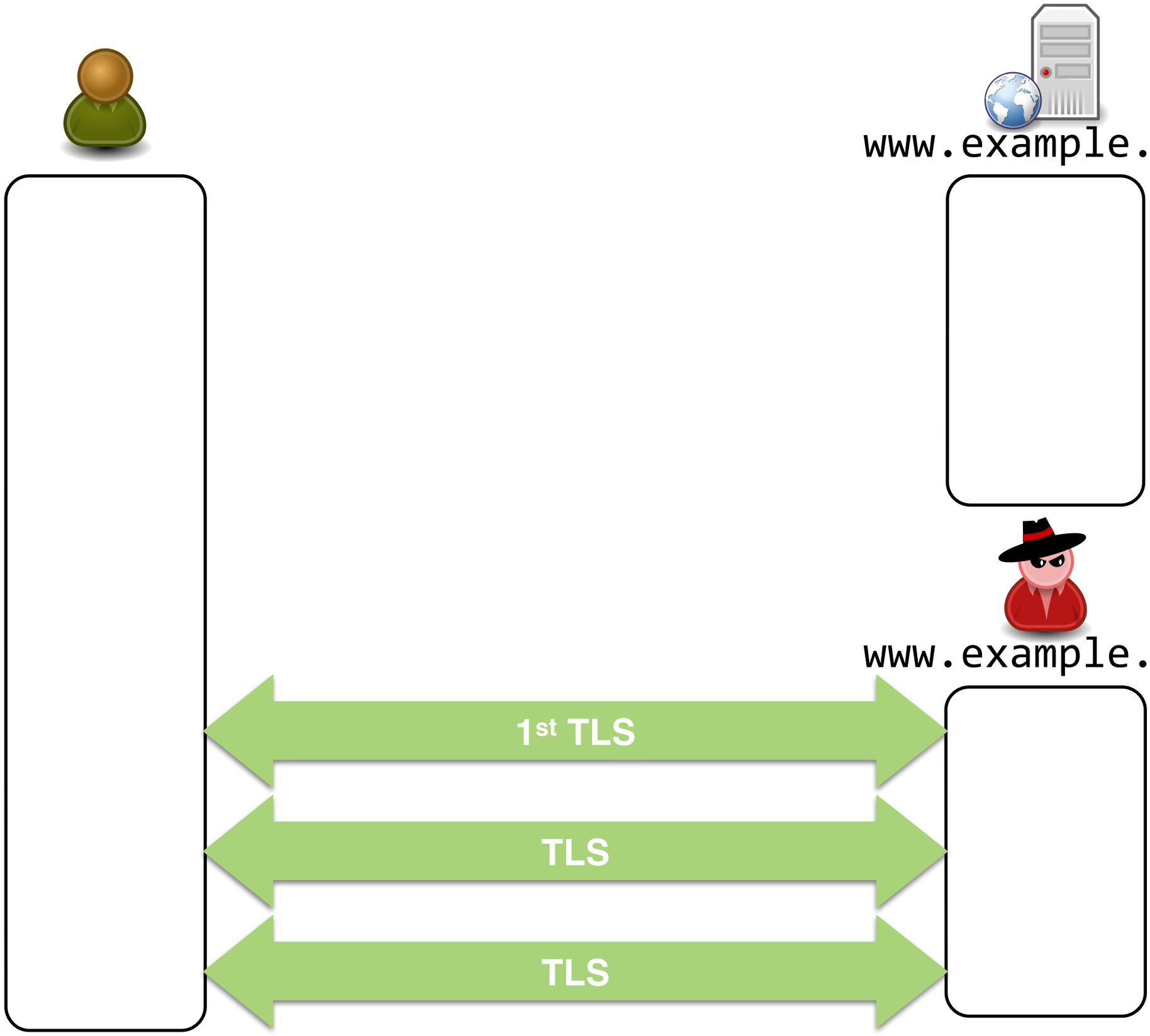
This Invariance is Enforced by the Browser



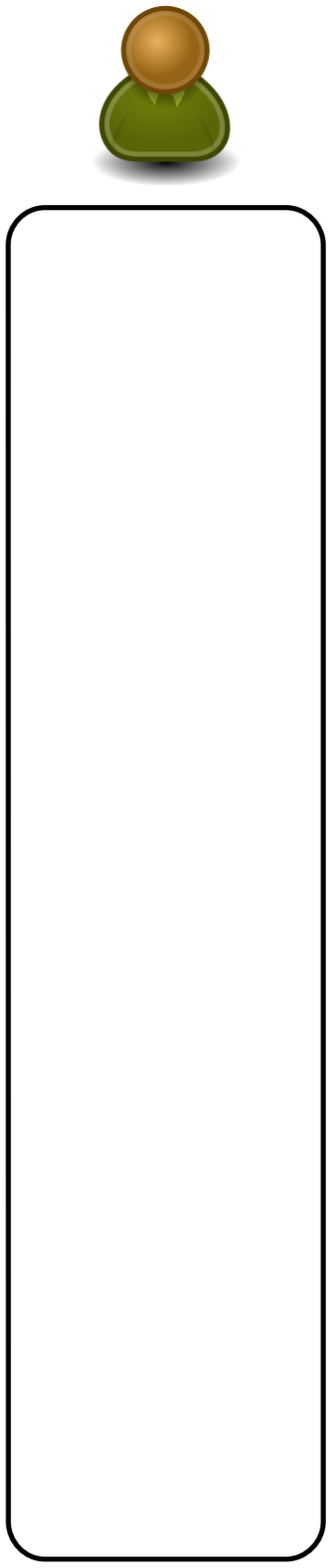


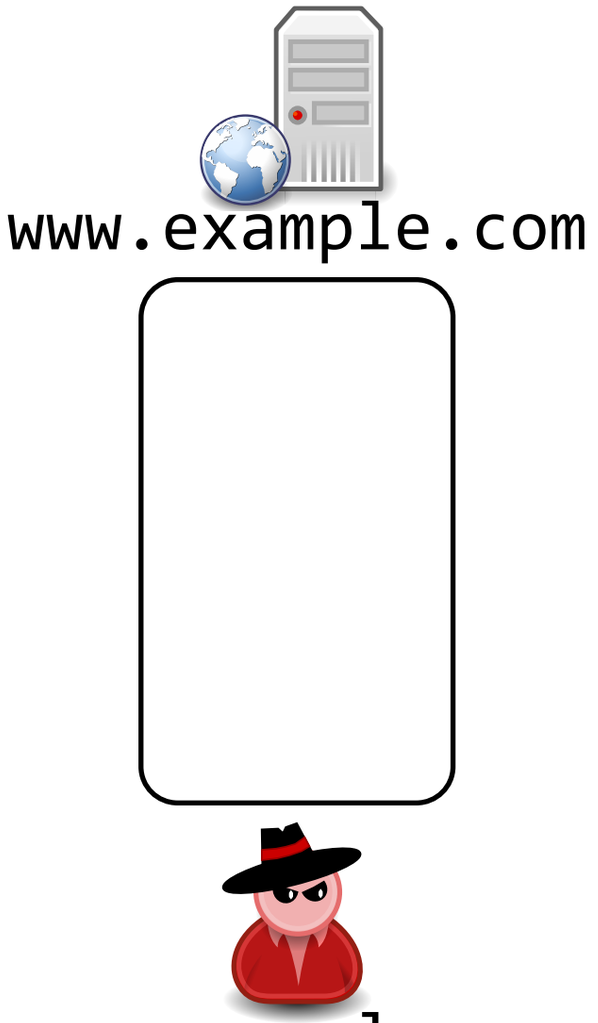


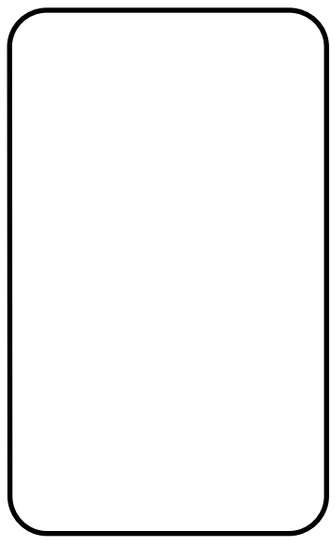
This Invariance is Enforced by the Browser



This Invariance is Enforced by the Browser







This Invariance is Enforced by the Browser

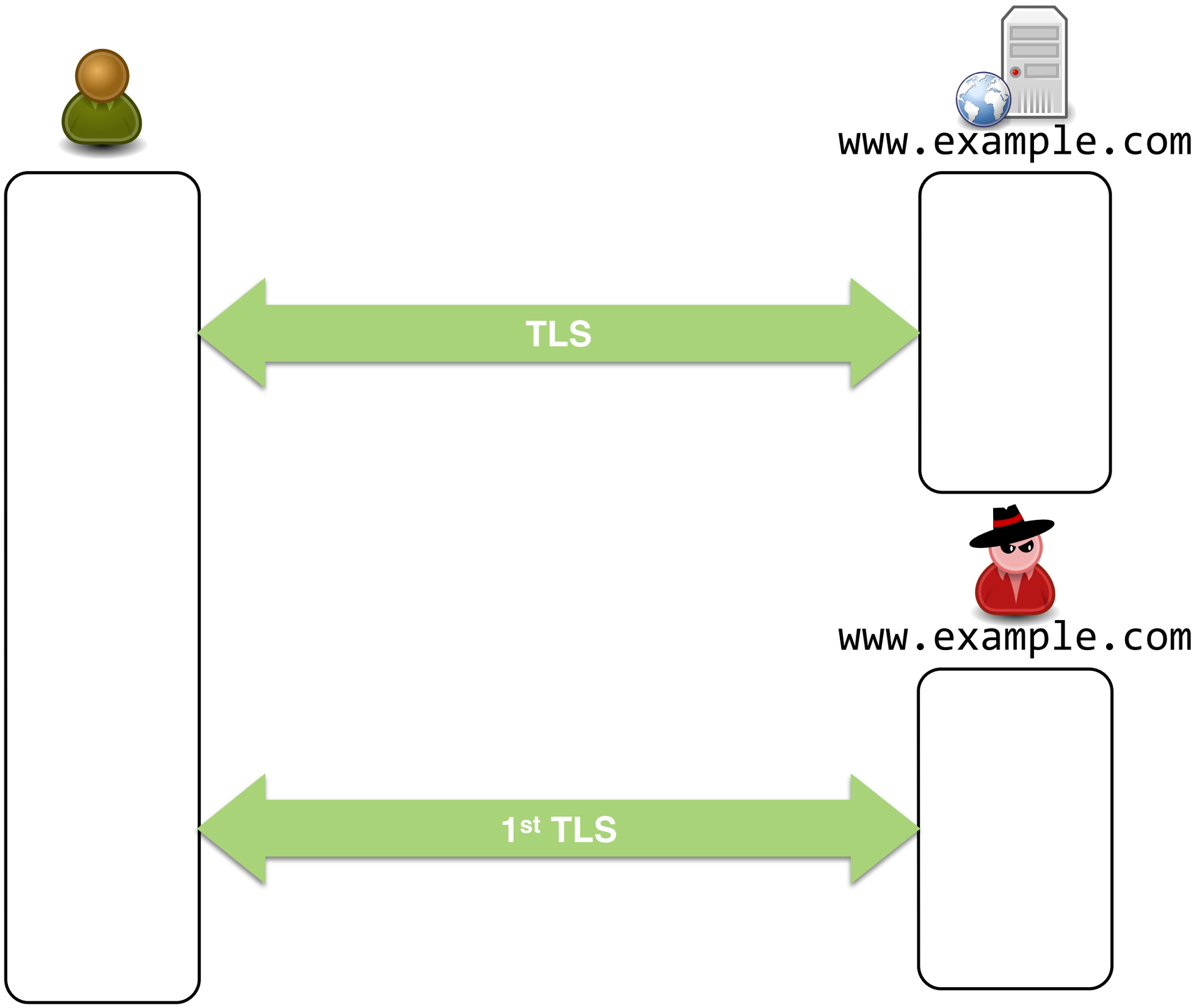
www.example.com

1st TLS

www.example.com

TLS

This Invariance is Enforced by the Browser



Example of Realizing Server Invariance

1. **Initialization (first connection)**
2. **Invariance verification**

www.example.com

Example of Realizing Server Invariance

ETH zürich

1. Initialization (first connection)
2. Invariance verification

www.example.com

1

r_b

store: $[r_b, r_s]$

TLS

‘Init’, r_b →

← r_s

r_s

store: [ , r_b, r_s]

Example of Realizing Server Invariance

ETH zürich

1. Initialization (first connection)
2. Invariance verification

www.example.com

r_b

store: $[r_b, r_s]$

1

r_s

store: [ , r_b, r_s]

‘Init’, r_b →

← r_s

TLS

2

‘Verify’, r_b →

$r_s \stackrel{?}{=} r'_s$

← r'_s

TLS

lookup: r'_s *from* [ , r_b]

TLS MITM prevention

TLS MITM prevention

Weak client authentication + Server authentication

passwords, conventional HTTP cookies, OTP, …

certificate pinning, certificate transparency, …

TLS MITM prevention

Weak client authentication + Server authentication

passwords, conventional HTTP cookies, OTP, …

certificate pinning, certificate transparency, …

Strong client authentication + *Server invariance*

Channel ID-based (FIDO U2F, channel-bound cookies),…

TLS MITM prevention

Weak client authentication + Server authentication

passwords, conventional HTTP cookies, OTP, …

certificate pinning, certificate transparency, …

Strong client authentication + *Server invariance*

Channel ID-based (FIDO U2F, channel-bound cookies),…

SISCA

Summary

Summary

- In web, servers can ask clients to execute arbitrary code
  - needs to be taken into account in protocol and system analysis

- TLS Channel IDs vulnerable to MITM-SITB attacks

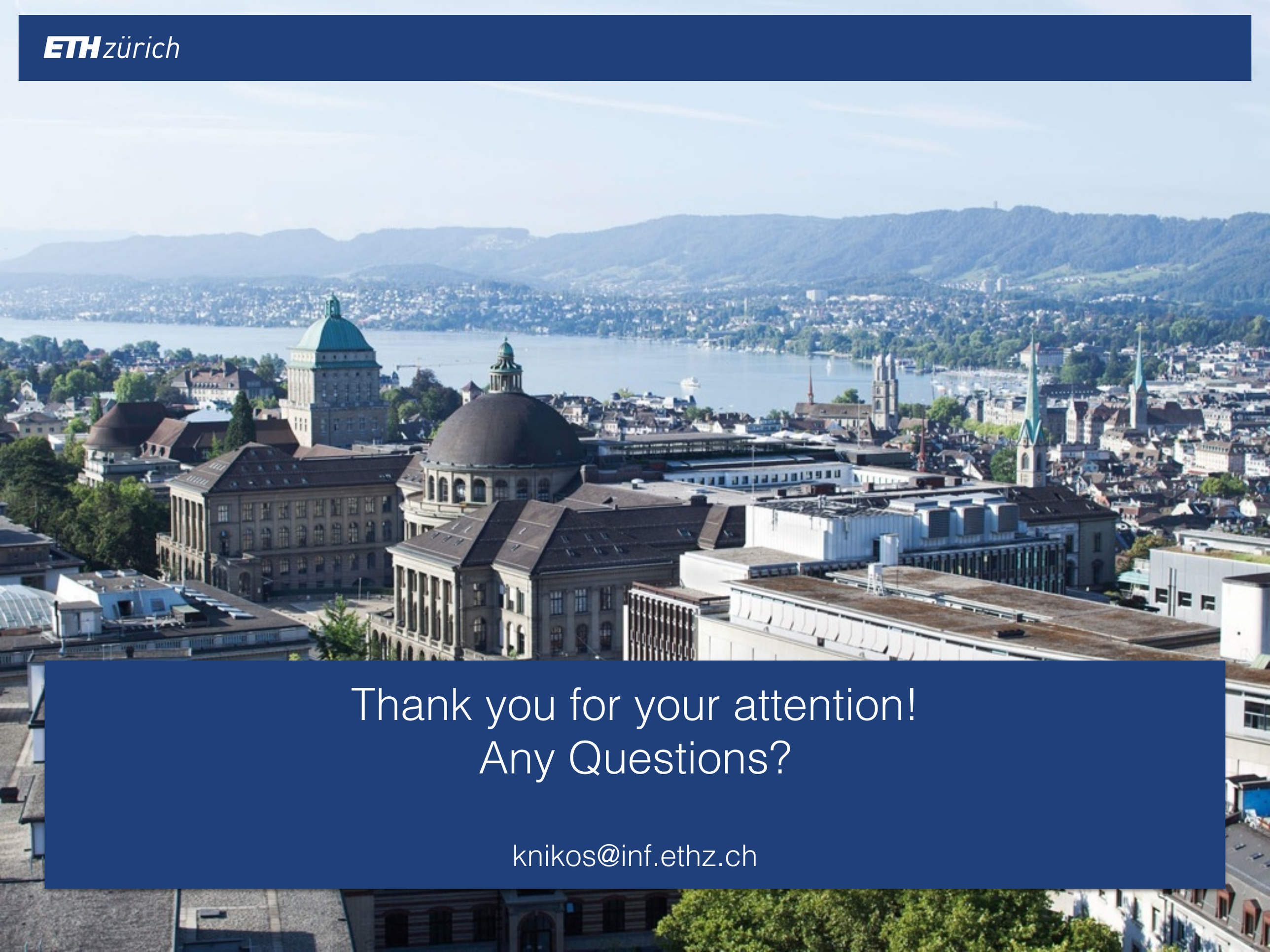
Summary

- In web, servers can ask clients to execute arbitrary code
  - needs to be taken into account in protocol and system analysis

- TLS Channel IDs vulnerable to MITM-SITB attacks

- To prevent MITM attacks we need either:
  - server authentication *or…*
  - **server invariance with Channel ID-based client authentication**

- Server invariance is easier to achieve than server authentication
  => we propose SISCA: Server Invariance with Strong Client Authentication



Thank you for your attention!
Any Questions?

knikos@inf.ethz.ch