Collective Certificate Management

Robin Berguerand

School of Computer and Communication Sciences
Decentralized and Distributed Systems lab

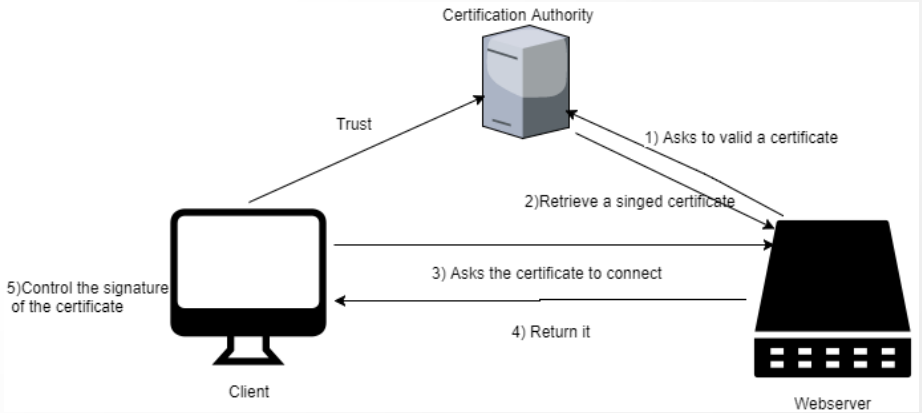
Bachelor Project

Responsible
Prof. Bryan Ford
EPFL / DEDIS

Supervisor
Philipp Jovanovic
EPFL / DEDIS
Linus Gasser
EPFL / DEDIS

Introduction

- Certificate
  - File that links a public key with its owner
  - Permits a secure internet connection
- Certification Authorities (CAs)
  - Validate and sign certificates
  - Must be trusted by all parties

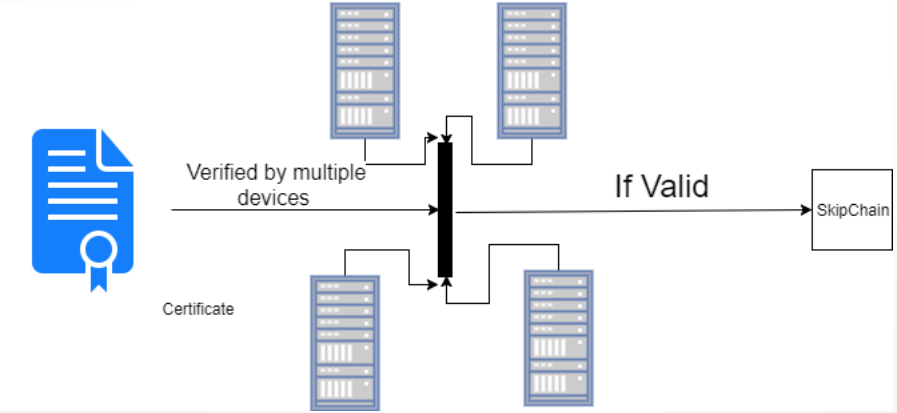


Certification Authority Issue

- Issue:
  - CA can validate fake certificates (even intentionally !)
- Consequences:
  - Impersonation of web server
  - Spying on Communications
- Main Problem : **Centrality**
  - Only one CA verifies a certificate

Solution

- **A collective Process:**
  - Multiple entities decide together if certificates are considered as valid using a **voting process**
  - The valid certificates are put in an irreversible storage to show that they were verified
  - Any modification on valid certificates should be collectively approved



The Project Tool

- Use **SkipChain** to store valid certificates:
  - Equivalent of a BlockChain
  - Decentralised because every participant has a copy of it
- Use **Cisc** to manage a SkipChain
  - The app permits to create it and to store any type of data in it
  - Permits multiple devices to connect to it
  - Implements a **voting** system
- This project add functionalities to Cisc for managing certificates

New Functionalities

- Additions to Cisc
  - Management of Certificates
    - Request
    - Store
    - List
    - Retrieve
    - Renew
    - Revoke

```
USAGE:
   SSH keystore client cert command [command options] [arguments...]

COMMANDS:
     request, q    request a certificate to letsencrypt and store it to the skipchain
     add, a        add a key/cert pair
     verify, v     verify the certificate against the root certificate
     renew, u      renew a certificate
     list, l       list the certificate
     revoke, k     revoke and delete a certificate
     retrieve, r   retrieve the certificate of a given key
```

  - Adaptation of the voting process

Certificates Life Cycle in Cisc

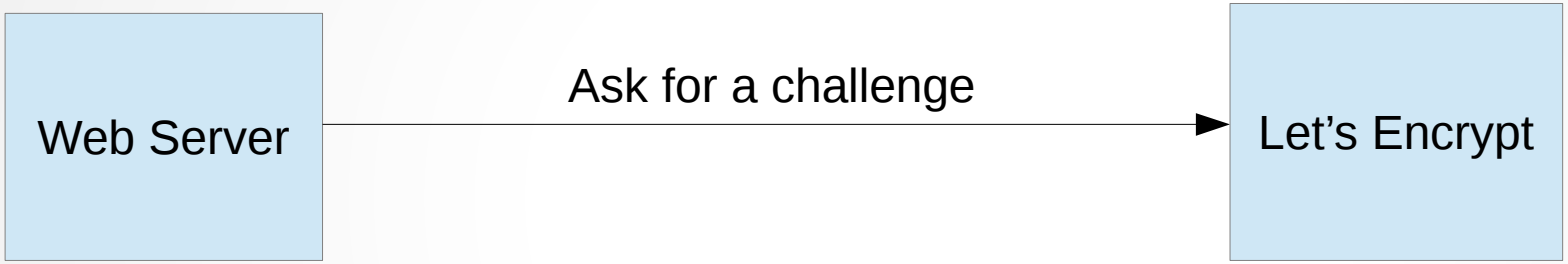
1) **Request it**
2) Verify it
3) Return it to the user
4) Submit it to vote
5) If it is validated, store it in a SkipChain
6) It can be managed inside the SkipChain

Let’s Encrypt CA

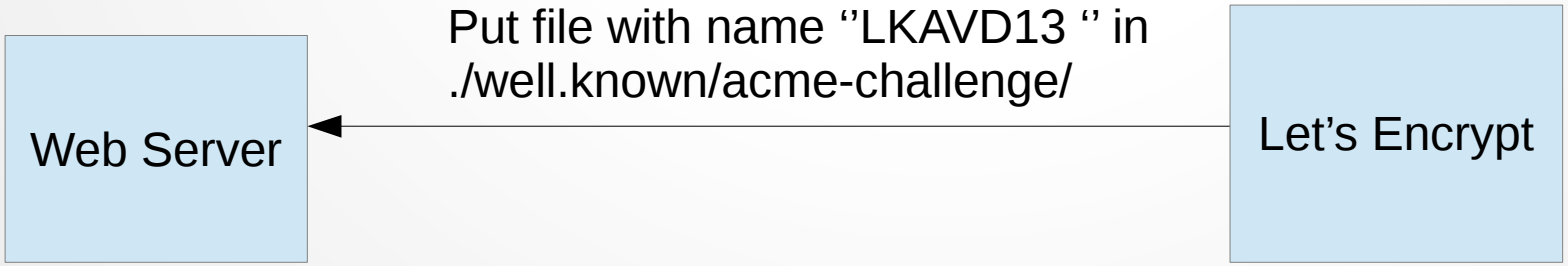
- Provides free certificates to domain servers
- Automated certificates generation
- Ensure matching between certificates and requester by using a **domain validation**

Automatic Domain Validation

- The client asks for a challenge

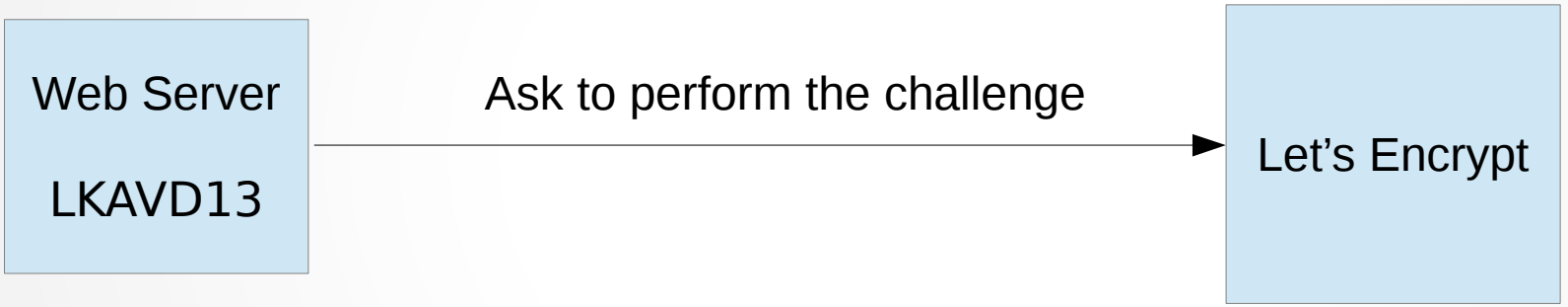


- Let’s Encrypt asks the client to put a file on a given place and with a given name

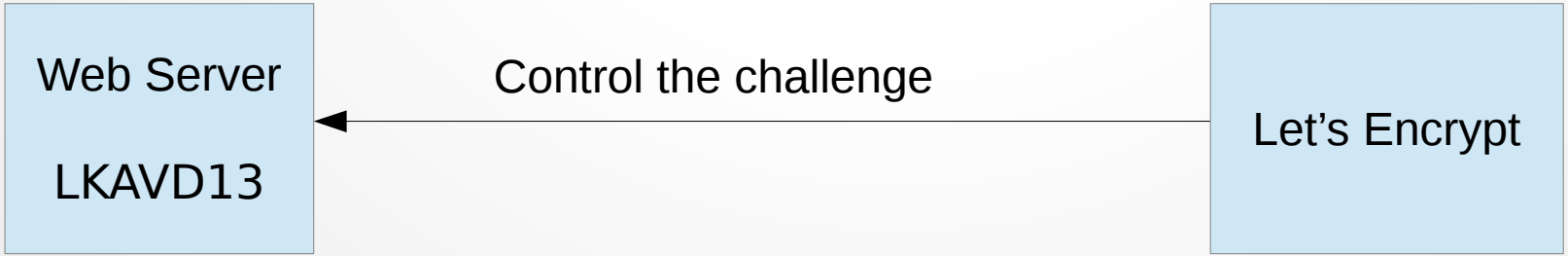


Automatic Domain Validation

- When the client is ready, it asks the ACME to perform the challenge

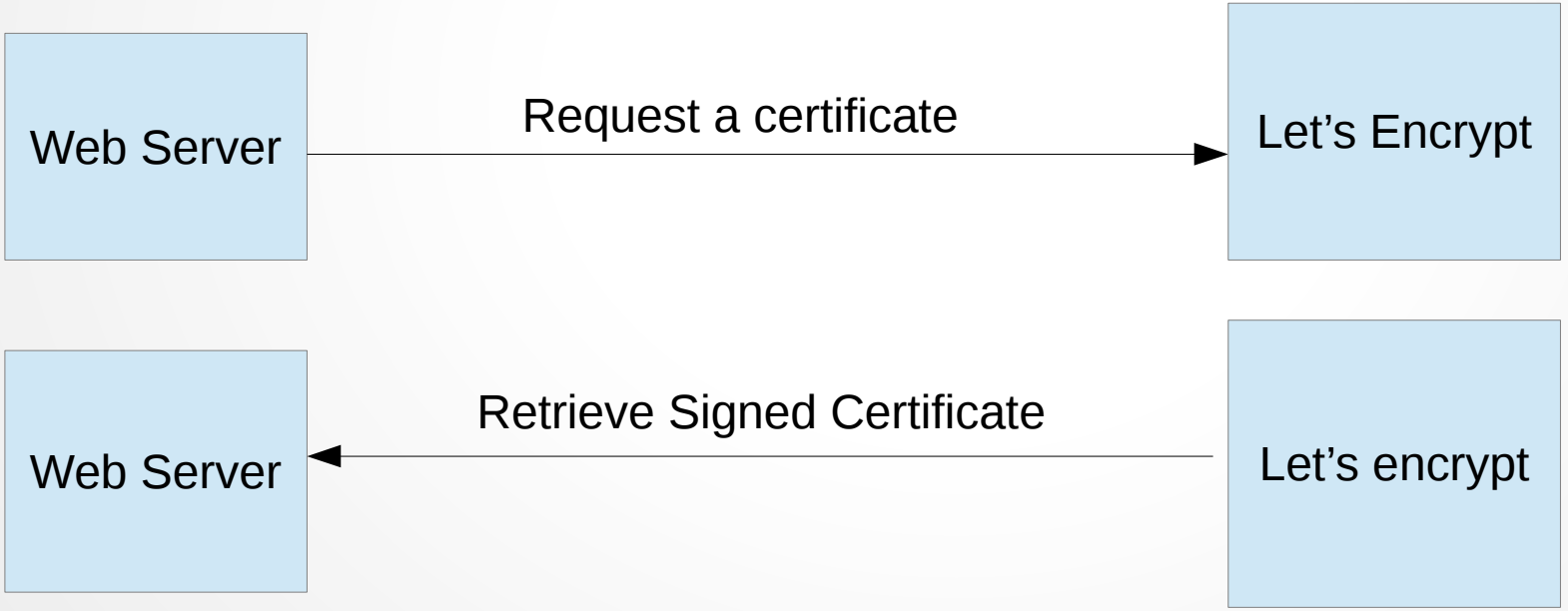


- Let's Encrypt controls the challenge by accessing the content and valid or invalid the challenge



Request Certificate

- Now that the web server is verified, it can request a certificate from Let’s Encrypt using a certificate request.



Certificates Life Cycle in Cisc

1) Request it
2) **Verify it**
3) Return it to the user
4) Submit it to vote
5) If it is validated, store it in a SkipChain
6) It can be managed inside the SkipChain

Verification

- Used to be sure that the certificate received is valid
  - Prevent issues during the requesting part.
- Chain to Let’s encrypt Root Certificate
  - Control that the new certificate is correctly signed
- Check multiple parameters
  - Validity date, domain name...

Certificates Life Cycle in Cisc

1) Request it
2) Verify it
3) **Return it to the user**
   - The certificate file and its corresponding keypair is returned
4) Submit it to vote
5) If it is validated, store it in a SkipChain
6) It can be managed inside the SkipChain

Certificates Life Cycle in Cisc

1) Request it
2) Verify it
3) Return it to the user
4) **Submit it to vote**
5) If it is validated, store it in a SkipChain
6) It can be managed inside the SkipChain

Voting Adaptation

- Every User connected to a SkipChain must vote on a new or an updated certificate for adding it permanently
- Before the vote, the application shows following information to the users about the certificate:
  - Its validity
  - The modification in case of an update
  - The whole certificate file.

Certificates Life Cycle in Cisc

1) Request it
2) Verify it
3) Return it to the user
4) Submit it to vote
5) **If it is validated, store it in a SkipChain**
    - A new SkipBlock containing the certificate is created
    - The certificate is now considered as collectively valid
6) It can be managed inside the SkipChain

Certificates Life Cycle in Cisc

1) Request it
2) Verify it
3) Return it to the user
4) Submit it to vote
5) If it is validated, store it in a SkipChain
6) **It can be managed inside the SkipChain**

Manage it

- Renew Certificate
  - Increase the validity period of a certificate
- Retrieve Certificate
  - Copy a certificate on the client devices
- Revoke Certificate
  - Revoke and delete a certificate
- List Certificate
  - See what is currently in the SkipChain

Demonstration

Advantages/Limitations

- Certificates can be considered as more trustworthy
- Harder to attack
- Free and easy management of certificates

- Users should still trust the whole protocol
- User should be heterogeneous enough to prevent group attack
- May take more time until a certificate is considered as valid

Future Works

- Connection
  - The browser must connect to the Cisc to obtain a web server’s certificate
- Warning
  - Web servers could be automatically warned when an attempt is made to submit a fake certificate corresponding to its domain name
- Automaticity
  - Automatic voting and renew/revoke system