Practical Issues with TLS Client Certificate Authentication

Arnis Parsovs

February 26, 2014

Motivation

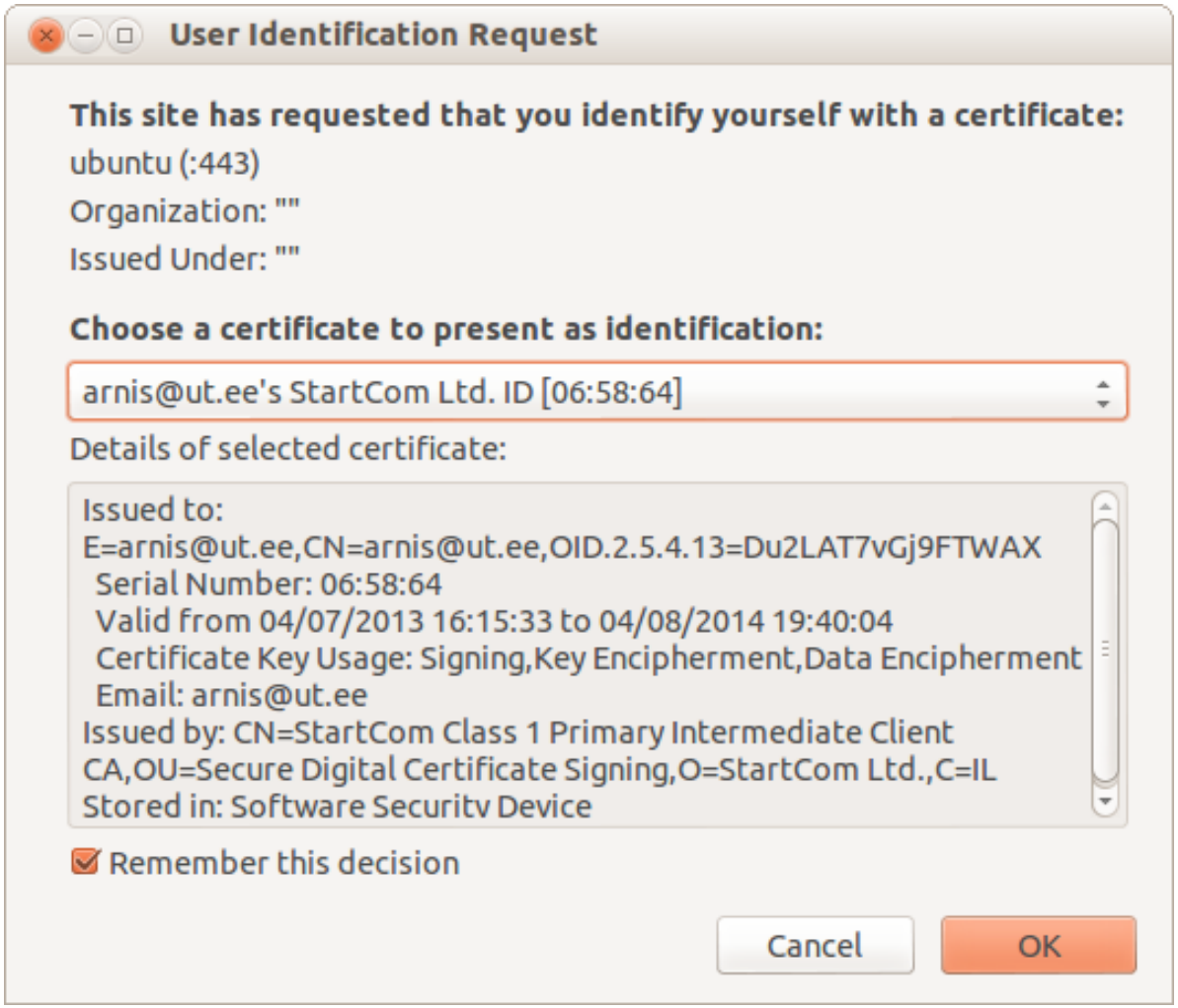
Problems with password authentication:

- Weak passwords
- Password reuse
- Insecure storage on server side
- Phishing attacks
- MITM attacks

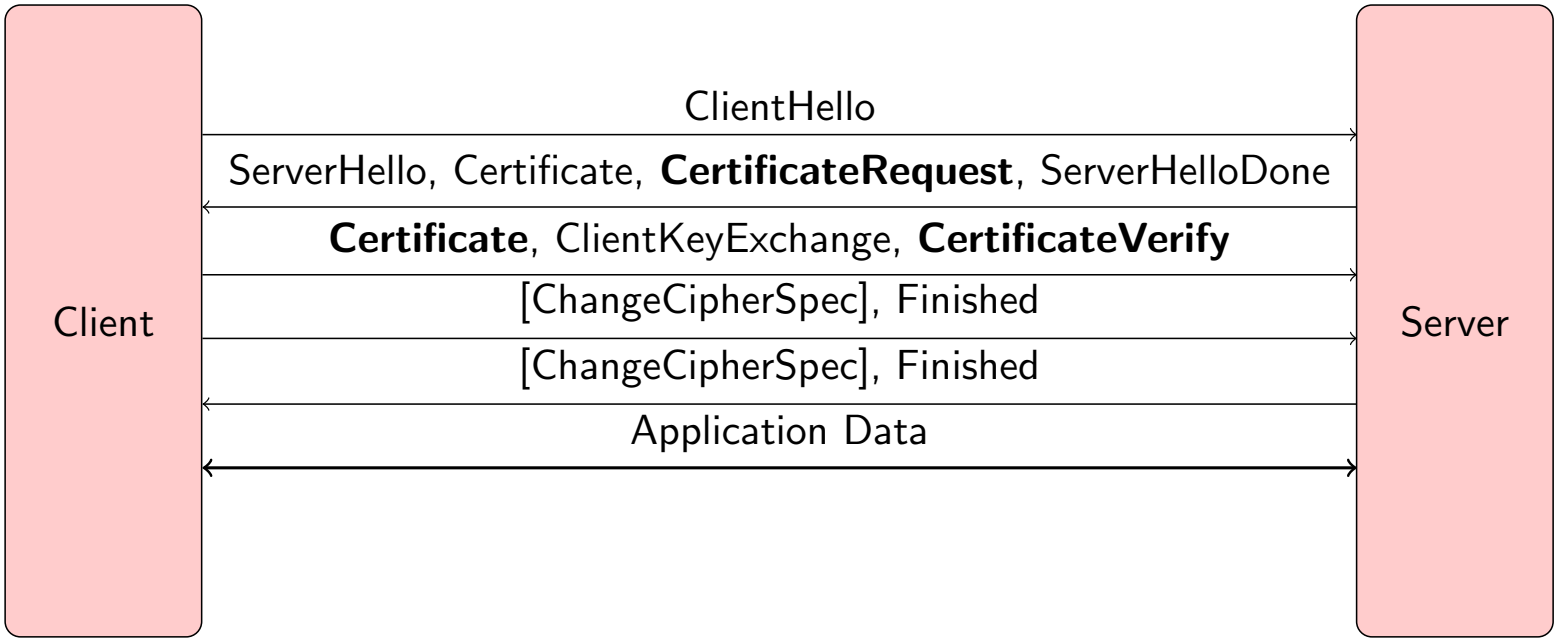
Solution to these problems – public key authentication in a form of **TLS Client Certificate Authentication (CCA)**

Supported by all major browsers!

TLS Client Certificate Authentication



TLS Client Certificate Authentication



- Private key has much better entropy than passwords
- The same certificate can be reused for different services
- No risk if server-side public key database leaks
- Private key cannot be phished by traditional phishing attacks
- MITM attacker (e.g., rogue CA) cannot impersonate the user
- No trusted third party required (!)

Estonia and TLS CCA

- Mandatory ID cards since 2002
- Two RSA key pairs:
  - For Qualified Digital Signatures
  - For TLS Client Certificate Authentication
- TLS CCA supported by all major e-service providers
  - Authentication to e-health services only by TLS CCA
  - Required to authorize online banking transactions >200 EUR

Research Objectives

What are the practical issues concerning TLS CCA deployment?
What should be improved on client and server side?

On server side:

- Apache mod_ssl (branch 2.2)

On client side:

- Mozilla Firefox (version 19.0)
- Google Chrome (version 25.0)
- Microsoft Internet Explorer (version 9.0)

Perform study on Estonian TLS CCA deployments.

Measurement Study of Estonian TLS CCA Deployments

- Analyzed 87 public service providers:

Software	**Hosts**	**Percent**
Apache mod_ssl	65	74.7%
MS IIS	10	11.5%
BigIP	4	4.6%
Oracle AS	3	3.4%
Tomcat	1	1.1%
Nginx	1	1.1%
Jetty	1	1.1%
unknown	2	2.3%

- 33% request certificate unencrypted
- 93% do not bind session to certificate
- 47% have superfluous CAs in trust store
- 45% have larger chain verification depth than needed
- 18% do not perform revocation checks

Things to Improve on Client Side (Browsers)

- Opt-in for strong locked same-origin policy
  - To isolate content served by MITM and legitimate connection
- JavaScript API in order to:
  - clear TLS session cache (reauthenticate)
  - clear client certificate selection (logout)
- Prevent deadlock in case CCA fails (Firefox, IE)
- Show warning if CCA requested on initial negotiation
- Client certificate selection window improvement:
  - Remember last client certificate choice

Things to Improve on Server Side (Apache mod_ssl)

- Provide session resumption support for CCA sessions
  - Important when CCA is performed by a smart card
- Implement flexible "`SSLVerifyClient require_any`"
  - To perform certificate verification at the application level
  - To provide personalized error messages in case of CCA failure
- Provide to environment variable the timestamp of CCA
  - To enforce the freshness of the proof of possession
- Provide better CCA audit trail

Conclusion

- Solution for secure user identity is already here
- Estonian example shows that it works in practice
- There are things to improve on client and server side
- Improvements do not require changes to the protocol

Thank you!