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!
Motivation

Problems with password authentication:

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

Problems with password authentication:

- Weak passwords
Motivation

Problems with password authentication:

- Weak passwords
- Password reuse

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

Supported by all major browsers!
Motivation

Problems with password authentication:

- Weak passwords
- Password reuse
- Insecure storage on server side

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

Supported by all major browsers!
Motivation

Problems with password authentication:

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

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

Supported by all major browsers!
Motivation

Problems with password authentication:

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

This site has requested that you identify yourself with a certificate:
ubuntu (443)
Organization: ""
Issued Under: ""

Choose a certificate to present as identification:

arnis@ut.ee's StartCom Ltd. ID [06:58:64]

Details of selected certificate:
Issued to:
E=arnis@ut.ee,CN=arnis@ut.ee,OID.2.5.4.13=Du2LAT7vGj9FTWAX
Serial Number: 06:58:64
Valid from 04/07/2013 16:15:33 to 04/08/2014 19:40:04
Certificate Key Usage: Signing, Key Encipherment, Data Encipherment
Email: arnis@ut.ee
Issued by: CN=StartCom Class 1 Primary Intermediate Client
CA,OU=Secure Digital Certificate Signing,O=StartCom Ltd.,C=IL
Stored in: Software Security Device

☑ Remember this decision
TLS Client Certificate Authentication

ClientHello
ServerHello, Certificate, CertificateRequest, ServerHelloDone
Certificate, ClientKeyExchange, CertificateVerify
[ChangeCipherSpec], Finished
[ChangeCipherSpec], Finished
Application Data

- 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 (!)
TLS Client Certificate Authentication

Client

ServerHello, Certificate, **CertificateRequest**, ServerHelloDone

**Certificate**, ClientKeyExchange, **CertificateVerify**

[ChangeCipherSpec], Finished

[ChangeCipherSpec], Finished

Application Data

Server

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 (!)
TLS Client Certificate Authentication

- Private key has much better entropy than passwords
TLS Client Certificate Authentication

- Private key has much better entropy than passwords
- The same certificate can be reused for different services
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
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
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
TLS Client Certificate Authentication

Client

ClientHello

ServerHello, Certificate, **CertificateRequest**, ServerHelloDone

**Certificate**, ClientKeyExchange, **CertificateVerify**

[ChangeCipherSpec], Finished

[ChangeCipherSpec], Finished

Application Data

Server

- 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
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
Estonia and TLS CCA

- Mandatory ID cards since 2002
Estonia and TLS CCA

• Mandatory ID cards since 2002
• Two RSA key pairs:
  • For Qualified Digital Signatures
  • For TLS Client Certificate Authentication
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
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
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.
Research Objectives

What are the practical issues concerning TLS CCA deployment?
Research Objectives

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

<table>
<thead>
<tr>
<th>Software Host</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache mod ssl</td>
<td>74.7%</td>
</tr>
<tr>
<td>MS IIS</td>
<td>11.5%</td>
</tr>
<tr>
<td>BigIP</td>
<td>4.6%</td>
</tr>
<tr>
<td>Oracle AS</td>
<td>3.4%</td>
</tr>
<tr>
<td>Tomcat</td>
<td>1.1%</td>
</tr>
<tr>
<td>Nginx</td>
<td>1.1%</td>
</tr>
<tr>
<td>Jetty</td>
<td>1.1%</td>
</tr>
<tr>
<td>unknown</td>
<td>2.3%</td>
</tr>
</tbody>
</table>

- 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
Measurement Study of Estonian TLS CCA Deployments

- Analyzed 87 public service providers:
Measurement Study of Estonian TLS CCA Deployments

- Analyzed 87 public service providers:

<table>
<thead>
<tr>
<th>Software</th>
<th>Hosts</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache mod_ssl</td>
<td>65</td>
<td>74.7%</td>
</tr>
<tr>
<td>MS IIS</td>
<td>10</td>
<td>11.5%</td>
</tr>
<tr>
<td>BigIP</td>
<td>4</td>
<td>4.6%</td>
</tr>
<tr>
<td>Oracle AS</td>
<td>3</td>
<td>3.4%</td>
</tr>
<tr>
<td>Tomcat</td>
<td>1</td>
<td>1.1%</td>
</tr>
<tr>
<td>Nginx</td>
<td>1</td>
<td>1.1%</td>
</tr>
<tr>
<td>Jetty</td>
<td>1</td>
<td>1.1%</td>
</tr>
<tr>
<td>unknown</td>
<td>2</td>
<td>2.3%</td>
</tr>
</tbody>
</table>

- 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
Measurement Study of Estonian TLS CCA Deployments

- Analyzed 87 public service providers:

<table>
<thead>
<tr>
<th>Software</th>
<th>Hosts</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache mod_ssl</td>
<td>65</td>
<td>74.7%</td>
</tr>
<tr>
<td>MS IIS</td>
<td>10</td>
<td>11.5%</td>
</tr>
<tr>
<td>BigIP</td>
<td>4</td>
<td>4.6%</td>
</tr>
<tr>
<td>Oracle AS</td>
<td>3</td>
<td>3.4%</td>
</tr>
<tr>
<td>Tomcat</td>
<td>1</td>
<td>1.1%</td>
</tr>
<tr>
<td>Nginx</td>
<td>1</td>
<td>1.1%</td>
</tr>
<tr>
<td>Jetty</td>
<td>1</td>
<td>1.1%</td>
</tr>
<tr>
<td>unknown</td>
<td>2</td>
<td>2.3%</td>
</tr>
</tbody>
</table>

- 33% request certificate unencrypted
Measurement Study of Estonian TLS CCA Deployments

- Analyzed 87 public service providers:

<table>
<thead>
<tr>
<th>Software</th>
<th>Hosts</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache mod_ssl</td>
<td>65</td>
<td>74.7%</td>
</tr>
<tr>
<td>MS IIS</td>
<td>10</td>
<td>11.5%</td>
</tr>
<tr>
<td>BigIP</td>
<td>4</td>
<td>4.6%</td>
</tr>
<tr>
<td>Oracle AS</td>
<td>3</td>
<td>3.4%</td>
</tr>
<tr>
<td>Tomcat</td>
<td>1</td>
<td>1.1%</td>
</tr>
<tr>
<td>Nginx</td>
<td>1</td>
<td>1.1%</td>
</tr>
<tr>
<td>Jetty</td>
<td>1</td>
<td>1.1%</td>
</tr>
<tr>
<td>unknown</td>
<td>2</td>
<td>2.3%</td>
</tr>
</tbody>
</table>

- 33% request certificate unencrypted
- 93% do not bind session to certificate
Measurement Study of Estonian TLS CCA Deployments

• Analyzed 87 public service providers:

<table>
<thead>
<tr>
<th>Software</th>
<th>Hosts</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache mod_ssl</td>
<td>65</td>
<td>74.7%</td>
</tr>
<tr>
<td>MS IIS</td>
<td>10</td>
<td>11.5%</td>
</tr>
<tr>
<td>BigIP</td>
<td>4</td>
<td>4.6%</td>
</tr>
<tr>
<td>Oracle AS</td>
<td>3</td>
<td>3.4%</td>
</tr>
<tr>
<td>Tomcat</td>
<td>1</td>
<td>1.1%</td>
</tr>
<tr>
<td>Nginx</td>
<td>1</td>
<td>1.1%</td>
</tr>
<tr>
<td>Jetty</td>
<td>1</td>
<td>1.1%</td>
</tr>
<tr>
<td>unknown</td>
<td>2</td>
<td>2.3%</td>
</tr>
</tbody>
</table>

• 33% request certificate unencrypted
• 93% do not bind session to certificate
• 47% have superfluous CAs in trust store
Measurement Study of Estonian TLS CCA Deployments

- Analyzed 87 public service providers:

<table>
<thead>
<tr>
<th>Software</th>
<th>Hosts</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache mod_ssl</td>
<td>65</td>
<td>74.7%</td>
</tr>
<tr>
<td>MS IIS</td>
<td>10</td>
<td>11.5%</td>
</tr>
<tr>
<td>BigIP</td>
<td>4</td>
<td>4.6%</td>
</tr>
<tr>
<td>Oracle AS</td>
<td>3</td>
<td>3.4%</td>
</tr>
<tr>
<td>Tomcat</td>
<td>1</td>
<td>1.1%</td>
</tr>
<tr>
<td>Nginx</td>
<td>1</td>
<td>1.1%</td>
</tr>
<tr>
<td>Jetty</td>
<td>1</td>
<td>1.1%</td>
</tr>
<tr>
<td>unknown</td>
<td>2</td>
<td>2.3%</td>
</tr>
</tbody>
</table>

- 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
Measurement Study of Estonian TLS CCA Deployments

- Analyzed 87 public service providers:

<table>
<thead>
<tr>
<th>Software</th>
<th>Hosts</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache mod_ssl</td>
<td>65</td>
<td>74.7%</td>
</tr>
<tr>
<td>MS IIS</td>
<td>10</td>
<td>11.5%</td>
</tr>
<tr>
<td>BigIP</td>
<td>4</td>
<td>4.6%</td>
</tr>
<tr>
<td>Oracle AS</td>
<td>3</td>
<td>3.4%</td>
</tr>
<tr>
<td>Tomcat</td>
<td>1</td>
<td>1.1%</td>
</tr>
<tr>
<td>Nginx</td>
<td>1</td>
<td>1.1%</td>
</tr>
<tr>
<td>Jetty</td>
<td>1</td>
<td>1.1%</td>
</tr>
<tr>
<td>unknown</td>
<td>2</td>
<td>2.3%</td>
</tr>
</tbody>
</table>

- 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 Client Side (Browsers)

- Opt-in for strong locked same-origin policy
Things to Improve on Client Side (Browsers)

- Opt-in for strong locked same-origin policy
  - To isolate content served by MITM and legitimate connection
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 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)
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)
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)
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
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:
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 environment variable the timestamp of CCA to enforce the freshness of the proof of possession.
• Provide better CCA audit trail.
Things to Improve on Server Side (Apache mod_ssl)

- Provide session resumption support for CCA sessions
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
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”
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
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
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
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
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!
Conclusion

- Solution for secure user identity is already here
Conclusion

- Solution for secure user identity is already here
- Estonian example shows that it works in practice
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
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
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!