Hardening Persona: Improving Federated Login on the Web

Michael Dietz
and
Dan S. Wallach
Motivation

X.com

TLS-OBC
PhoneAuth
Motivation
Motivation
Motivation
Existing federated login protocols

- SAML
- OpenID
- Persona (BrowserId)
- OAuth
- OAuth2 / OpenIdConnect
- Kerberos
Persona Overview

X.com

Initial Login

Session Login

Relying Party

User

Laptop
Persona Overview

Initial Login

Session Login

X.com

Relying Party

Log me in with Persona
Persona Overview

Initial Login

X.com

Session Login

What’s the users email address?

Log me in with Persona

Relying Party
Persona Overview

Please login

I'm alice@X.com

Relying Party
Persona Overview

Initial Login

X.com

User is alice@X.com

Session Login

Relying Party
Persona Overview

Initial Login

X.com

Session Login

Relying Party

User is alice@X.com
Persona Overview

Initial Login

X.com

Session Login

Relying Party

Cookie: alice@X.com
Two areas for attack

• MITM the connection between user and RP
  - Replay identity assertions

• Steal relying party cookie after login
Identity assertion theft

Initial Login

X.com

User is Alice@X

Session Login

Relying Party
Identity assertion theft

Initial Login

X.com

User is Alice@X

MITM

Session Login

Relying Party
Identity assertion theft

Initial Login

Session Login

X.com

Relying Party

User is Alice@X

MITM
Two areas for attack

• MITM the connection between user and RP
  - Replay identity assertions

• Steal relying party cookie after login
RP cookie theft

X.com

User is Alice@X

This is Alice

Relying Party
RP cookie theft

Initial Login

X.com

Session Login

User is Alice@X

MITM

Relying Party

This is Alice
RP cookie theft

Initial Login

Session Login

X.com

Relying Party

This is Alice

MITM
Extensions to Persona
Design Goals

• Strengthen identity assertions against MITM theft

• Allow relying parties to establish a key for communication with the user
Initial Assumptions

Identity Provider

TLS-OBC PhoneAuth

K_{BI}

Relying Party
Initial Assumptions

- Identity Provider
  - Alice: $K_B$

- TLS-OBC
  - PhoneAuth

- Relying Party
Initial Assumptions

Identity Provider
Alice: $K_{BI}$

Relying Party

TLS

$K_{BI}$  $K_{BR}$
Initial Assumptions

Identity Provider
Alice: $K_{BI}$

Relying Party
$K_{BR}$

TLS
Persona-OBC-Central

• Uses thePersona underpinnings, works more like OAuth2
  • IDP sees RP’s public key
  • Can track user logins to RPs
  • Simple to implement
Goal

Identity Provider  Alice: $K_{BI}$

Alice’s browser controls

$K_{BI}$  $K_{BR}$

Relying Party
Post Key API
Post Key API

• **Goal**: Convince IDP that browser controls two OBCs used on two different domains
Post Key API

- **Goal:** Convince IDP that browser controls two OBCs used on two different domains

- Creates cross certification between two origin bound certificate keys
Post Key API

• **Goal**: Convince IDP that browser controls two OBCs used on two different domains

• Creates cross certification between two origin bound certificate keys

• API exposed as browser extension
Post Key API

• **Goal**: Convince IDP that browser controls two OBCs used on two different domains

• Creates cross certification between two origin bound certificate keys

• API exposed as browser extension
  • Similar to postMessage() call
Post Key API

![Diagram of Post Key API]

- **Target Origin**: \([K_A]_{K_B}, [K_B]_{K_A}\)
- **Source Origin**: `w1.postKey()`
- **Browser**: \(w_1\)
- **Identity Provider**: \(K_A\) and \(K_B\) belong to the same browser.
- **Relying Party**: \(K_B\)
- **TLS**: \(K_A, [K_A]_{K_B}, [K_B]_{K_A}\)
Post Key API

• Assumptions
  - IDP received cross cert on TLS channel associated with $K_A$
  - IDP knows $K_A$ is a key Alice’s browser controls
Post Key API

• Assumptions
  - IDP received cross cert on TLS channel associated with $K_A$
  - IDP knows $K_A$ is a key Alice’s browser controls

$$[K_A, A.com]_{K_B}, [K_B, B.com]_{K_A}$$
Post Key API

- Assumptions
  - IDP received cross cert on TLS channel associated with $K_A$
  - IDP knows $K_A$ is a key Alice’s browser controls

$$[K_A, \text{A.com}]_{K_B}, [K_B, \text{B.com}]_{K_A}$$

Alice’s browser says $K_B$
Post Key API

• Assumptions
  - IDP received cross cert on TLS channel associated with $K_A$
  - IDP knows $K_A$ is a key Alice’s browser controls

Browser can sign with $K_B$

Alice’s browser says $K_B$

$[K_A, A.com]_{K_B}$, $[K_B, B.com]_{K_A}$
Goal

Identity Provider
Alice: $K_{BI}$

IDP says user is Alice and controls $K_{BR}$

Relying Party

$K_{BI}$  $K_{BR}$
Goal

Identity Provider: Alice: $K_{BI}$

IDP says user is Alice and controls $K_{BR}$

Relying Party
Goal

Identity Provider
Alice: $K_{BI}$

IDP says user is Alice and controls $K_{BR}$

Relying Party

TLS

$K_{BI}$ $K_{BR}$
Goal

Identity Provider
Alice: $K_{BI}$

Relying Party
IDP says user is Alice and controls $K_{BR}$

TLS

$K_{BI}$ $K_{BR}$
Goal
Goal

Identity Provider
Alice: $K_{BI}$

Relying Party
Alice: $K_{BR}$

TLS
Goal

Identity Provider

Alice: $K_{BI}$

Relying Party

Alice: $K_{BR}$

I believe $K_{BR}$ speaks for Alice
Goal

Identity Provider
Alice: $K_{BI}$

Relying Party
Alice: $K_{BR}$

I believe $K_{BR}$ speaks for Alice
Persona-OBC-Local:
Preserve Persona semantics
Persona Specifics

- IDP cannot track where the user logs in
- Uses public key crypto (in the browser)
  - IDP signs short lived browser key
  - Browser creates identity assertion with browser key
  - RP can verify assertions without an online IDP
Persona-OBC-Local

- IDP signs browser controlled key $K_B$ and user identity with its well known key $K_I$
- Browser creates identity assertions on the fly by signing new TLS-OBC key for RP with $K_B$
Persona-OBC-Local protocol

1. Browser sends cross certification and channel bound cookie to IDP

Browser

IDP.com

C_{user} Cert_{idp}

RP.com

Cert_{rp}

IDP.com

[K_B]_{K_B} + [K_B]_{K_BI}

TLS
Persona-OBC-Local protocol

1. Browser sends cross certification and channel bound cookie to IDP

Browser

IDP.com

[Cert\textsubscript{idp}]

RP.com

[Cert\textsubscript{rp}]

IDP.com

\([K_{BI}]_{KB} + [K_B]_{KBI}\)

\(C_{user}\)

TLS
Persona-OBC-Local protocol

2. IDP creates identity certificate
Persona-OBC-Local protocol

3. IDP sends identity certificate to browser for storage
Persona-OBC-Local protocol

3. IDP sends identity certificate to browser for storage
Persona-OBC-Local protocol

4. User wants to log into RP, Browser creates identity assertion

[User, K_B, T]_{KI} + [RP.com, K_R, T]_{KB}
Persona-OBC-Local protocol

4. User wants to log into RP, Browser creates identity assertion
Persona-OBC-Local protocol

5. RP verifies assertion

Browser

IDP.com

Cert_{idp}

C_{user}

RP.com

Cert_{rp}

IDP.com

[K]

C_{user}

RP.com

[User, K_{B}, T]_{KI} + [RP.com, K_{R}, T]_{KB}
Persona-OBC-Local protocol

5. RP verifies assertion

Browser

IDP.com

IDP.com

Cert_{idp}

C_{user}

RP.com

Cert_{rp}

IDP.com

[K]

C_{user}

RP.com

[User, K_B, T]_K_I + [RP.com, K_R, T]_K_B
Persona-OBC-Local protocol

5. RP verifies assertion

Browser

IDP.com

C_user

cert_idp

RP.com

cert_rp

IDP.com

[K]

User, K_B, T_{KI} + [RP.com, K_R, T_{KB}]

TLS

TLS
Persona-OBC-Local protocol

5. RP verifies assertion

![Diagram showing the Persona-OBC-Local protocol with changes from the initial version. The diagram includes Trusted Server, Browser, and RP.com with certificates and keys involved in the TLS handshake and assertion verification process. The keys and certificates are labeled as follows:

- IDP.com: [K]
- C_user: [C]
- Cert_{idp}: [Cert_{idp}]
- Cert_{rp}: [Cert_{rp}]
- User: [User, K_B, T]_{KI}
- RP.com: [RP.com, K_R, T]_{KB}

The diagram illustrates the flow of communication and encryption used in the protocol.]
Persona-OBC-Local protocol

5. RP mints (channel-bound) cookie for user
Persona-OBC-Local protocol

5. RP mints (channel-bound) cookie for user
Attack #1

- Attacker between browser and IDP
Attack #1

• Attacker between browser and IDP
Attack #2

• Attacker between browser and RP

Browser

IDP.com

[C_user] [Cert_idp]

RP.com

[Cert_rp]

TLS

IDP.com

[K]

RP.com

[User, K_B, T]_{K_I} + [RP.com, K_R, T]_{K_B}
Attack #2

- Attacker between browser and RP

Browser
- IDP.com
  - Cert_{idp}
  - C_{user}
- RP.com
  - Cert_{rp}

TLS
IDP.com [K
RP.com

[User, K_B, T]_{KI} + [RP.com, K_R, T]_{KB}
Attack #3

- Attacker between browser and RP
- Attacker impersonates browser

Browser

IDP.com

IDP.com

Browser

IDP.com

[User, K_B, T]_{KI} + [RP.com, K_R, T]_{KA}
Attack #3

- Attacker between browser and RP
- Attacker impersonates browser

Browser
- IDP.com
  - C_user
  - Cert_idp
- RP.com
  - Cert_rp

IDP.com

RP.com

TLS

[User, K_B, T]_{KI} + [RP.com, K_R, T]_{KA}
Attack #3

- Attacker between browser and RP
- Attacker impersonates browser
Protocol implementation

• Proof of concept IDP and RP implementations for Persona-OBC-Local
• Both written in Python
  - Use Nexus Authorization Logic proof checker to verify assertions
• BAN logic formalization of both protocols
  • Local and Central variants
Conclusion

• Two persona extensions
  • Better MITM protection for identity assertions

• Leverage channel between IDP and user to create channel between user and RP

• RP uses a different key than IDP to communicate with the user (for privacy)
Questions?
mdietz@gmail.com