Token Binding over HTTPS

Vinod Anupam

IETF 94 ● Yokohama ● November 2015
Overview

1. Recap (for newcomers)
2. Changes to tokbind-https
3. Threat model
Overview

1. Recap (for newcomers)
2. Changes to tokbind-https
3. Threat model
Recap: The Token Binding Header

provided_token_binding: {
    signature(EKM),
    public_key
}

GET / HTTP/1.1
Host: example.com
Token-Binding: DLF02LDSK3DMS28SA...
User-Agent: ...

...
Recap: The Token Binding Header

- client uses different key pairs for different servers
- client protects private keys
- client discloses public key of pair to server
- client proves to server that it controls private key
Example: Sending Header

- Client transmits Token Binding key

```plaintext
POST /Login HTTP/1.1
Token-Binding: DLF02LDSK3DMS28SA...
...
username=bob&pw=password
```
Example: Binding Cookies

- Server binds tokens to Token Binding key

```
POST /Login HTTP/1.1
Token-Binding: DLF02LDSK3DMS28SA...
username=bob&pw=password

Client example.com

200 OK
Set-Cookie: SID=sdkhfoeirusakjn34aslkd
...
<html>Welcome!</html>
```

```
cookie_val = {
    user_id: 1234456,
    last_login: 9348230984,
    tb_key: public_key
}
```
Example: Verifying Cookies

- Server confirms that cookie matches Token Binding key

```plaintext
GET /Inbox.html HTTP/1.1
Token-Binding: DLF02LDSK3DMS28SA...
Cookie: SID=sdkhfoeirusakinf34asldk

provided_token_binding: {
  signature(EKM),
  public_key example.com
}

cookie_val = {
  user_id: 1234456,
  last_login: 9348230984,
  tb_id: public_key example.com,
  ...
}
```
1: "I want to log in"
2: "Go get an id token from idp.com"
1: “I want to log in”

2: “Go get an id token from idp.com”

3: “Give me an id token for rp.com”

   “Who are you??”
   “username: bob & passwd: 12345”

4: “Here is your id token for rp.com”
1: “I want to log in”
2: “Go get an id token from idp.com”
3: “Give me an id token for rp.com”
4: “Here is your id token for rp.com”
5: “Here is the id token from idp.com”
6: “You’re logged in. Here is your cookie”

“Who are you??”
“username: bob & passwd: 12345”
1: “I want to log in”

2: “Go get an id token from idp.com”

3: “Give me an id token for rp.com”

4: “Here is your id token for rp.com”

5: “Here is the id token from idp.com”

6: “You’re logged in. Here is your cookie”

Without federation support, token will be bound to IDP

username: bob & passwd: 12345
Without federation support, RP can’t tell if client is legitimate user of token.

Without federation support, token will be bound to IDP.

Username: bob, passwd: 12345
1: “I want to log in”
2: “Go get an id token from idp.com”
3: “Give me an id token for rp.com”
4: “Here is your id token for rp.com”
5: “Here is the id token from idp.com”
6: “You’re logged in. Here is your cookie”

Need: Token from IDP must be bound to client-RP TLS connection.
1: “I want to log in”
2: “Go get an id token from idp.com”
3: “Give me an id token for rp.com”
4: “Here is your id token for rp.com”
5: “Here is the id token from idp.com”
6: “You’re logged in. Here is your cookie”

Need: Client must tell IDP the binding key for client-RP TLS connection

Need: Token from IDP must be bound to client-RP TLS connection.
How to Trigger Referred Token Bindings?

Relying Party uses HTTP Redirect

302 Moved Temporarily
Location: https://idp.com/rp-login
Include-Referer-Token-Binding-Id: true
How to Trigger Referred Token Bindings?

Relying Party uses HTTP Redirect

302 Moved Temporarily
Location: https://idp.com/rp-login
Include-Referer-Token-Binding-Id: true

RP tells client to send its RP token binding key to target server of redirect (IDP)
Federation with HTTP Redirects

GET / HTTP/1.1
Token-Binding: QWR26DLF02LDSK3DM...

302 Moved Temporarily
Location: https://idp.com/rp-login
Include-Referer-Token-Binding-Id: true
Federation with HTTP Redirects

GET / HTTP/1.1
Token-Binding: QWR26DLF02LDSK3DM…

302 Moved Temporarily
Location: https://idp.com/rp-login
Include-Referer-Token-Binding-Id: true

GET /rp-login HTTP/1.1
Host: idp.com
Token-Binding: MDLF02LDSK3DMS28S…
Referer: rp.com
User-Agent: …
Federation with HTTP Redirects

GET / HTTP/1.1
Token-Binding: QWR26DLF02LDSK3DMS28S...

GET /rp-login HTTP/1.1
Host: idp.com
Token-Binding: MDLF02LDSK3DMS28S...
Referer: rp.com
User-Agent: ...

provided_token_binding: {
  signature(EKM),
  public_key_idp.com
}

referred_token_binding: {
  signature(EKM),
  public_key_rp.com
}
Federated Binding

GET /rp-login HTTP/1.1
Token-Binding: DLF02LDSK3DMS28SA...
Cookie: SID=sdkhfoeirusakjnf34aslkd...

302 Moved Temporarily
Location: https://rp.com/login?tok=hfoeimk...

Client

tenon = {
    user_id: 1234456,
    name: Bob,
    tb_id: public_key
}

idp.com
Overview

1. Recap (for newcomers)
2. Changes to tokbind-https
3. Threat model
Header: Sec-Token-Binding ⇒ Token-Binding

Prove key possession by signing EKM (instead of tls_unique)
  - TLS Exported Keying Material, per RFC 5705

Updated Security Considerations
  - Why disallow scripts from setting Token-Binding header?
  - Why prove possession of two keys for federation?
Overview

1. Recap (for newcomers)
2. Changes to tokbind-https
3. Threat model
Intent of Token Binding

server verifies public-key signature in token binding ⟹ client controls corresponding private key
Intent of Token Binding

Why?
Binding token to public key should make it possible to enforce that token can be used only by a client that can prove possession of the private key, and by nobody else.

server verifies public-key signature in token binding $\xrightarrow{}$ client controls corresponding private key
Threats

1. Attacker uses victim’s private key
   - Countermeasure: Keep private key secret

2. Attacker makes victim present attacker’s public key
   (== client sends attacker-generated token-binding header)
Threats

1. Attacker uses victim’s private key
   - countermeasure: keep private key secret
   - countermeasure: never transmit private key over network

2. Attacker makes victim present attacker’s public key
   (client sends attacker-generated token-binding header)
Threats

1. Attacker uses victim’s private key
   ○ countermeasure: keep private key secret
   ○ countermeasure: never transmit private key over network

2. Attacker makes victim present attacker’s public key
   (client sends attacker-generated token-binding header)
   ○ countermeasure: keep EKM secret
Threats

1. Attacker uses victim’s private key
   ○ countermeasure: keep private key secret
   ○ countermeasure: never transmit private key over network

2. Attacker makes victim present attacker’s public key
   (client sends attacker-generated token-binding header)
   ○ countermeasure: keep EKM secret
   ○ countermeasure: don’t let attacker set Token-Binding header
# Threats

1. **Attacker uses victim’s private key**
   - countermeasure: keep private key secret
   - countermeasure: never transmit private key over network

2. **Attacker makes victim present attacker’s public key**
   (client sends attacker-generated token-binding header)
   - countermeasure: keep EKM secret
   - countermeasure: don’t let attacker set Token-Binding header
   - countermeasure: make client prove possession of every key in header

\[
\text{server verifies public-key signature in token binding} \quad \Rightarrow \quad \text{client controls corresponding private key}
\]
Questions