Token Binding over HTTPS

Dirk Balfanz

IETF 93 ● Prague ● July 2015

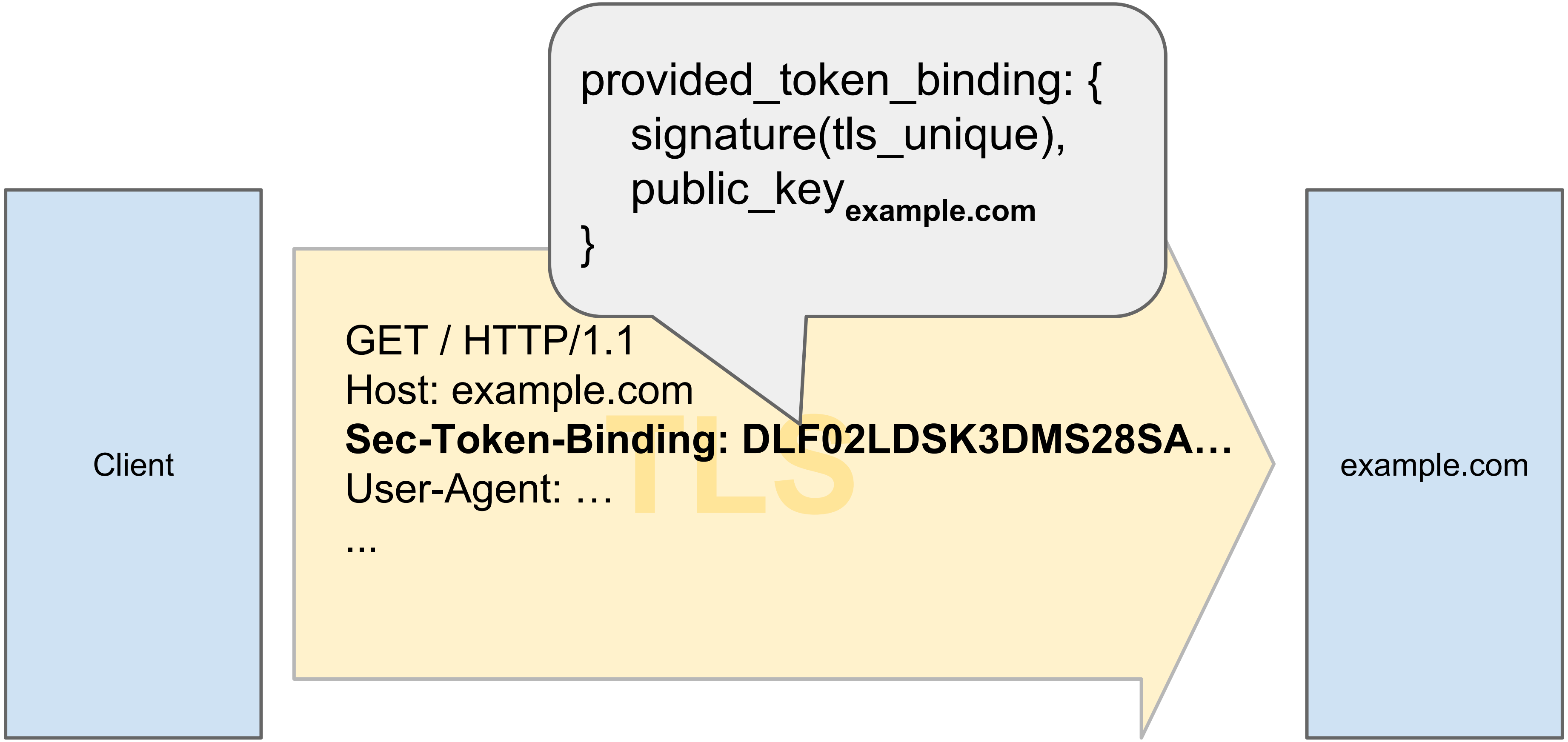
Overview

1. Recap (for newcomers)
2. Changes to tokbind-https
3. Threat model (or: Should there be a Sec- prefix?)
4. Requirements for federation

Overview

1. Recap (for newcomers)
2. Changes to tokbind-https
3. Threat model (or: Should there be a Sec- prefix?)
4. Requirements for federation

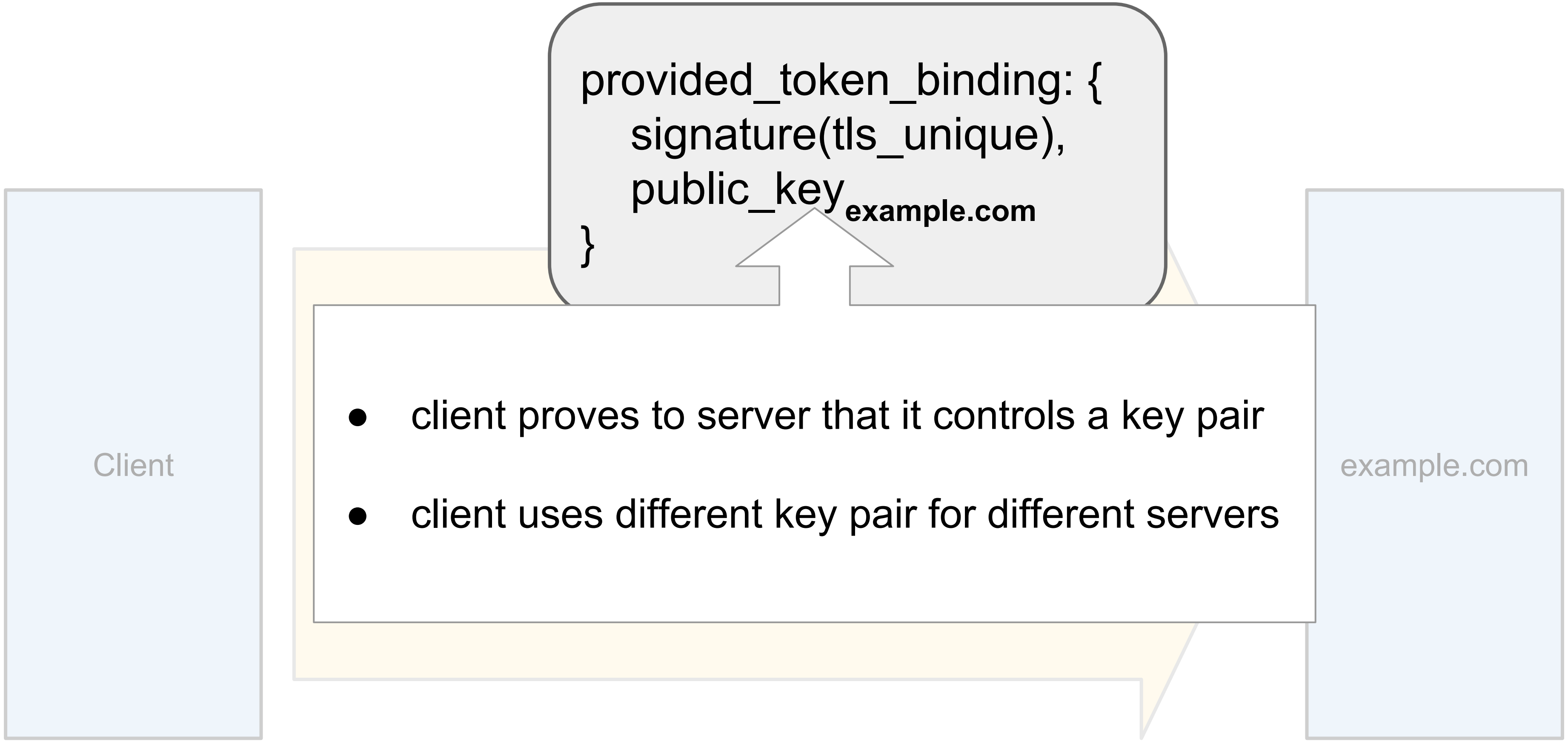
Recap: The Token Binding Header



```
provided_token_binding: {
    signature(tls_unique),
    public_key_example.com
}
```

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

Recap: The Token Binding Header

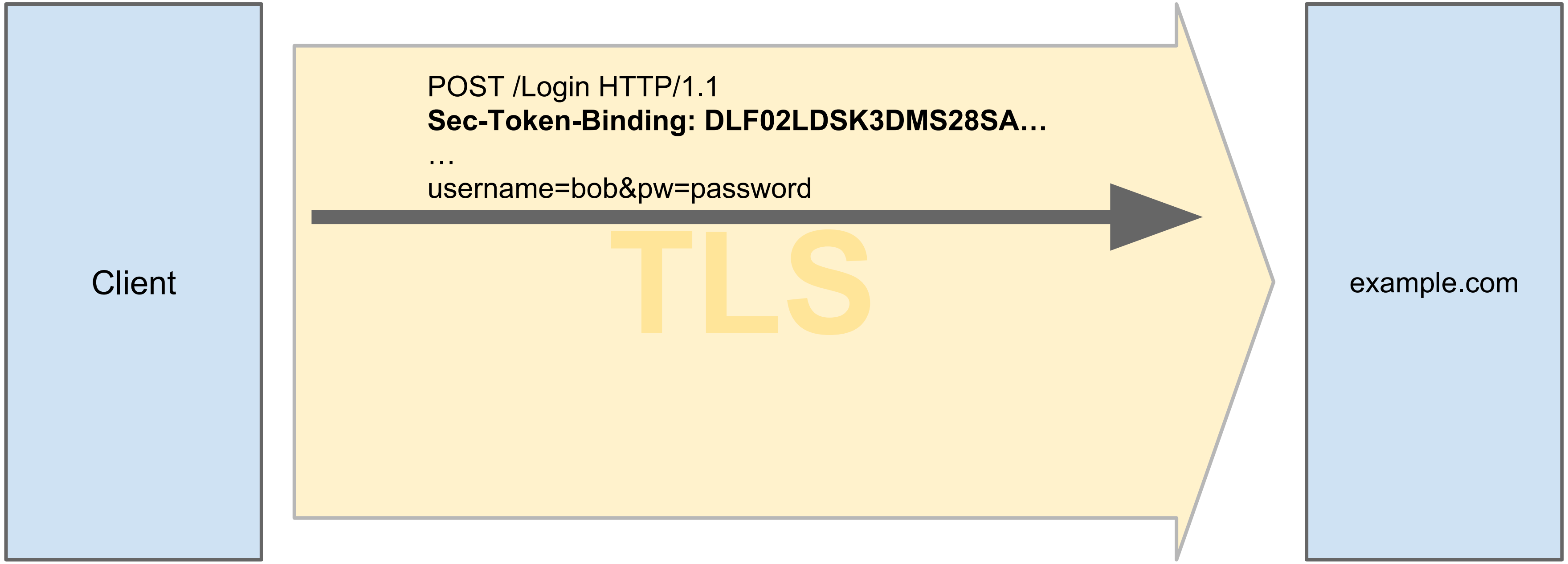


provided_token_binding: {
    signature(tls_unique),
    $\text{public\_key}_{\textbf{example.com}}$
}

- client proves to server that it controls a key pair
- client uses different key pair for different servers

Example: Binding Cookies

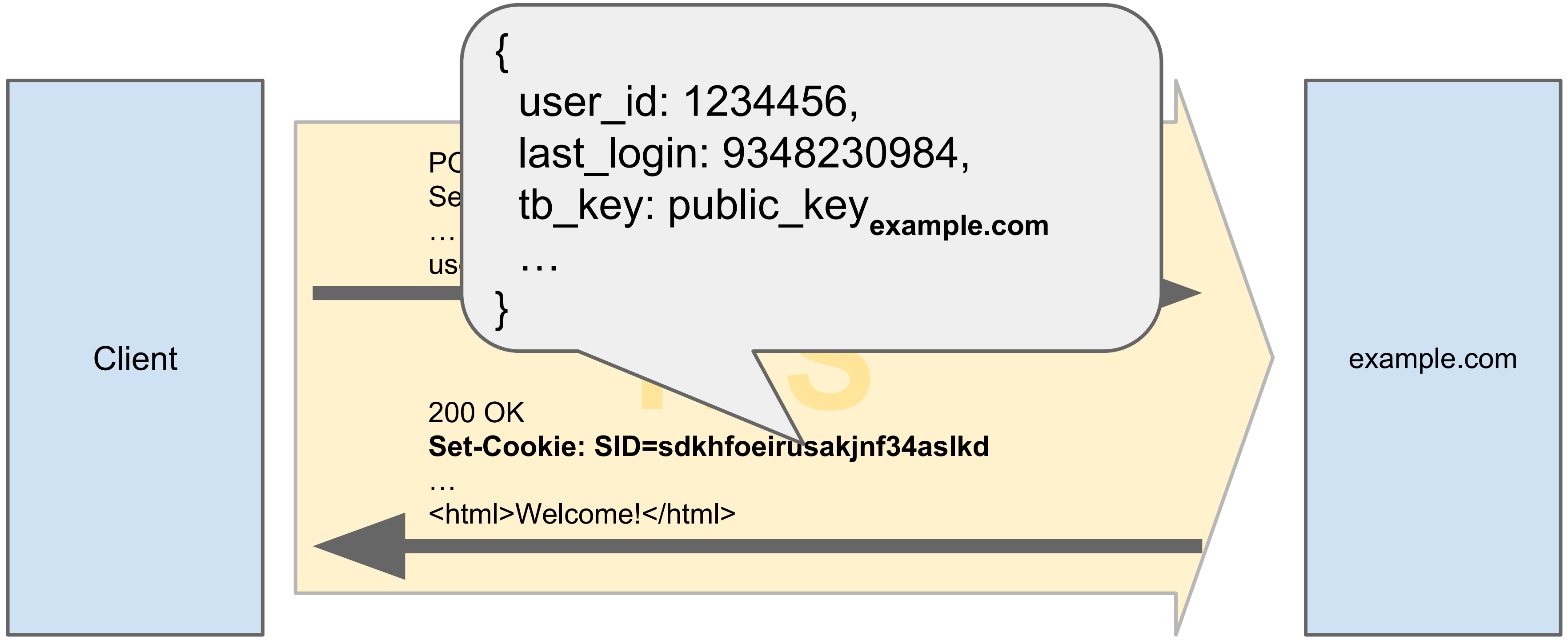
- Server can then bind tokens to Token Binding key



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

Example: Binding Cookies

- Server can then bind tokens to Token Binding key



```
{
  user_id: 1234456,
  last_login: 9348230984,
  tb_key: public_key_example.com
  …
}
```

Overview

1. Recap (for newcomers)
2. Changes to tokbind-https
3. Threat model (or: Should there be a Sec- prefix?)
4. Requirements for federation

diff tokbind-https-00 tokbind-https-01

- Header: Token-Binding ⇒ Sec-Token-Binding
- Clarification: what should you bind to?
  (public key, not Token Binding ID)
- Learned how to spell “true”
- Removed any mention of DOM APIs, which were meant to support federation
  (left the HTTP redirect approach in the document)

Overview

1. Recap (for newcomers)
2. Changes to tokbind-https
3. Threat model (or: Should there be a Sec- prefix?)
4. Requirements for federation

Intent

server verifies public-key signature in token binding $\Longrightarrow$ **client (and nobody else) controls corresponding private key**

(Why? Because binding a token to private key should ensure that the token can be used only by corresponding client, and nobody else.)

Threats

server verifies public-key signature in token binding $\Rightarrow$ **client (and nobody else) controls corresponding private key**

1. Attacker uses victim’s private key

2. Victim “uses” attacker’s private key
   (client is tricked into sending attacker-controlled token-binding header)

Threats

server verifies public-key signature in token binding $\Rightarrow$ **client (and nobody else) controls corresponding private key**

1. Attacker uses victim’s private key
   - countermeasure: keep private key secret
2. Victim “uses” attacker’s private key
   (client is tricked into sending attacker-controlled token-binding header)

Threats

server verifies public-key signature in token binding	$\Longrightarrow$	**client (and nobody else) controls corresponding private key**

1. Attacker uses victim’s private key
    - countermeasure: keep private key secret
2. Victim “uses” attacker’s private key
   (client is tricked into sending attacker-controlled token-binding header)
    - countermeasure: keep tls-unique secret
    - countermeasure: don’t let attacker set token-binding header

Threats

server verifies public-key signature in token binding $\Longrightarrow$ **client (and nobody else) controls corresponding private key**

1. Attacker uses victim's private key
   - countermeasure: keep private key secret

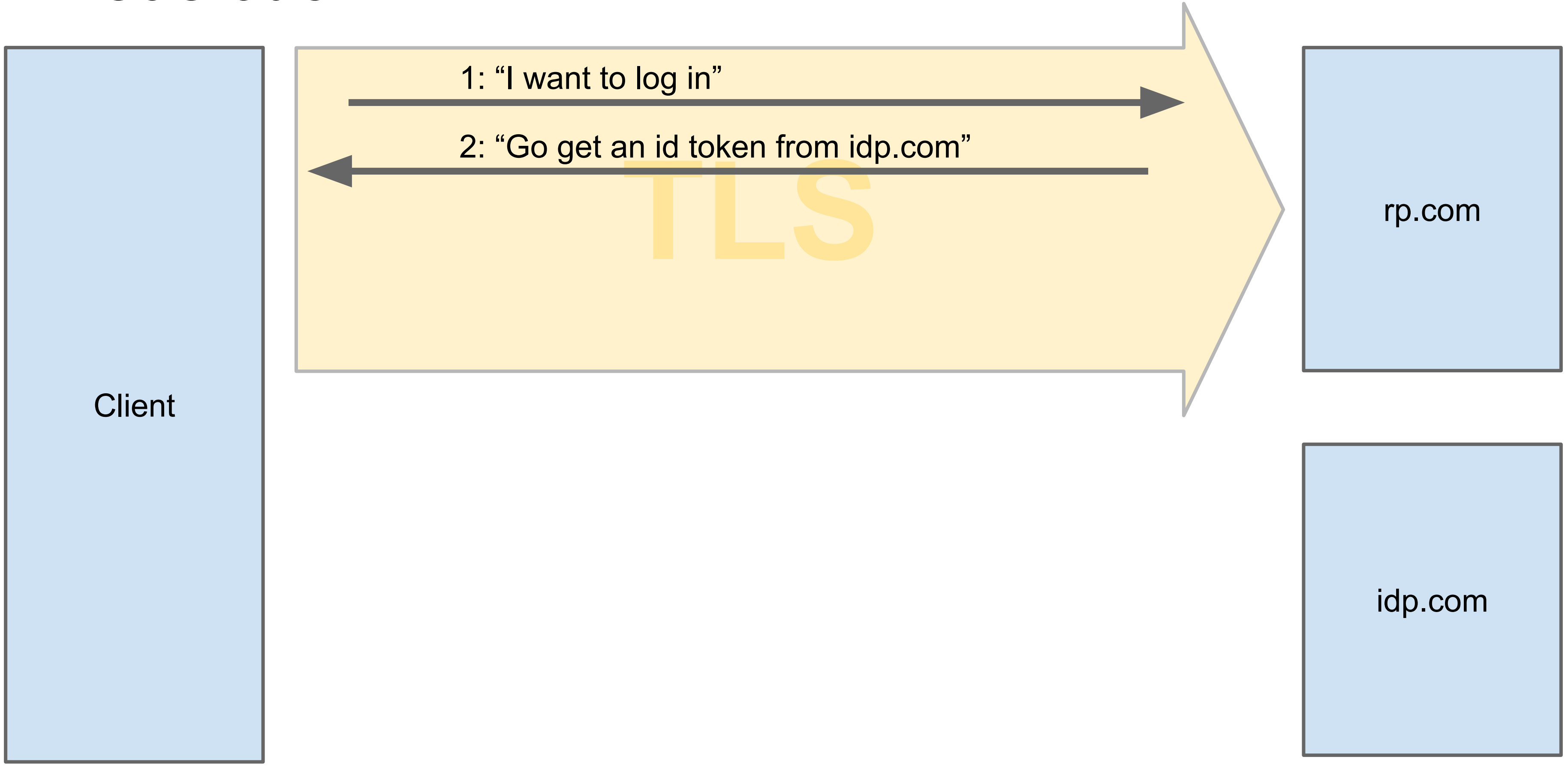
2. Victim "uses" attacker's private key
   (client is tricked into sending attacker-controlled token-binding header)
   - countermeasure: keep tls-unique secret
   - countermeasure: don't let attacker set token-binding header

Reason for Sec- prefix

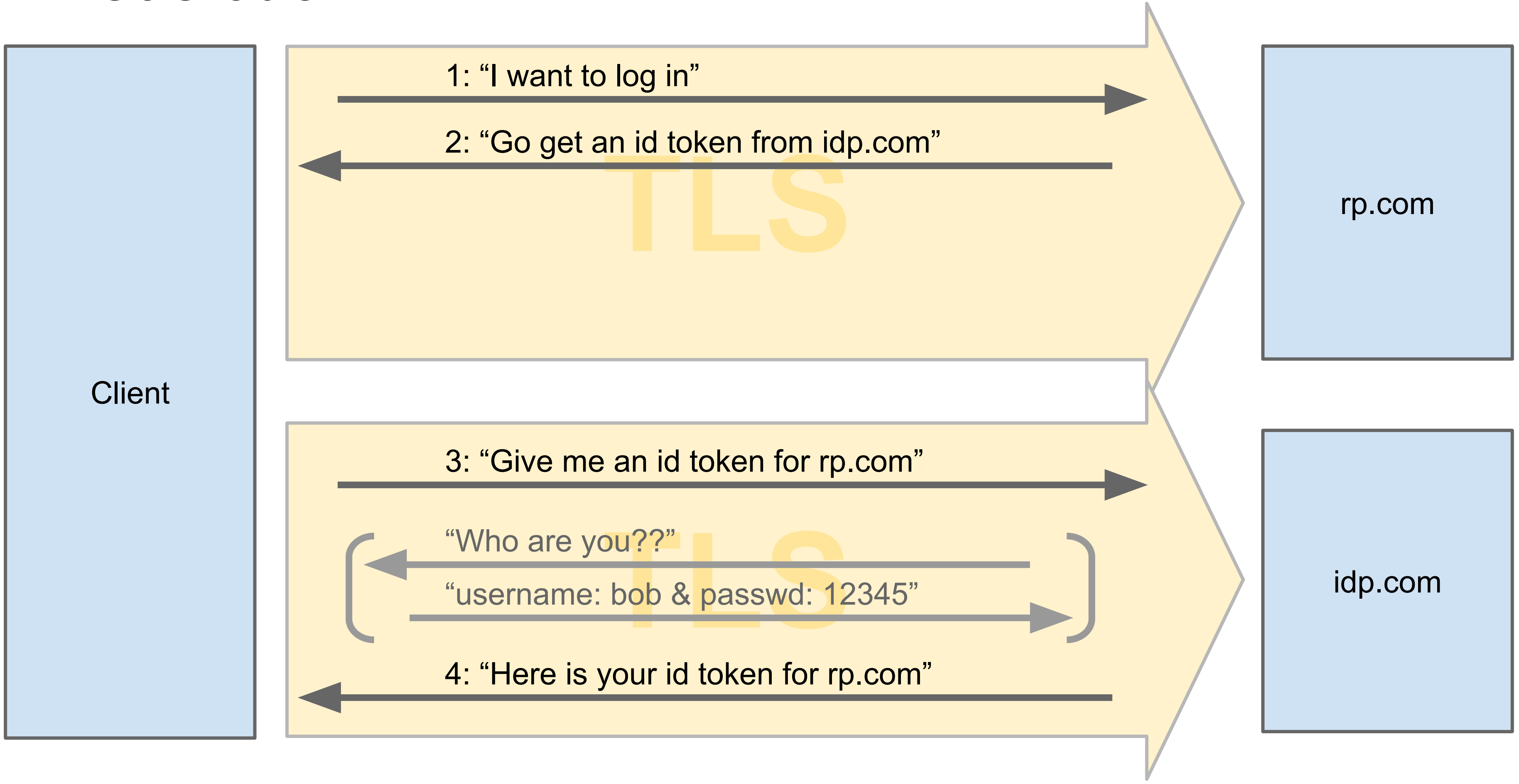
Overview

1. Recap (for newcomers)
2. Changes to tokbind-https
3. Threat model (or: Should there be a Sec- prefix?)
4. Requirements for federation

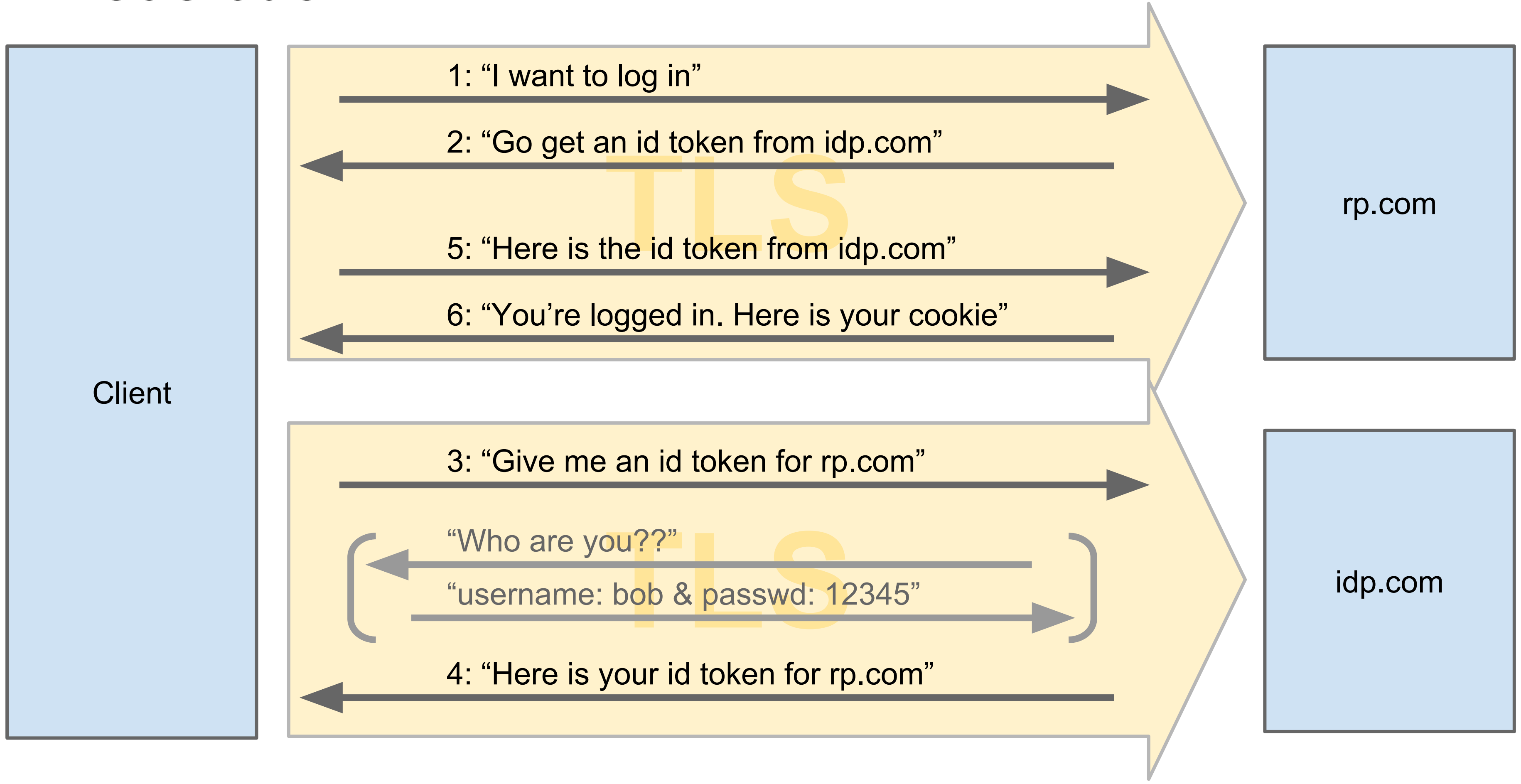
Federation


Client
1: “I want to log in”
2: “Go get an id token from idp.com”
TLS
rp.com
idp.com

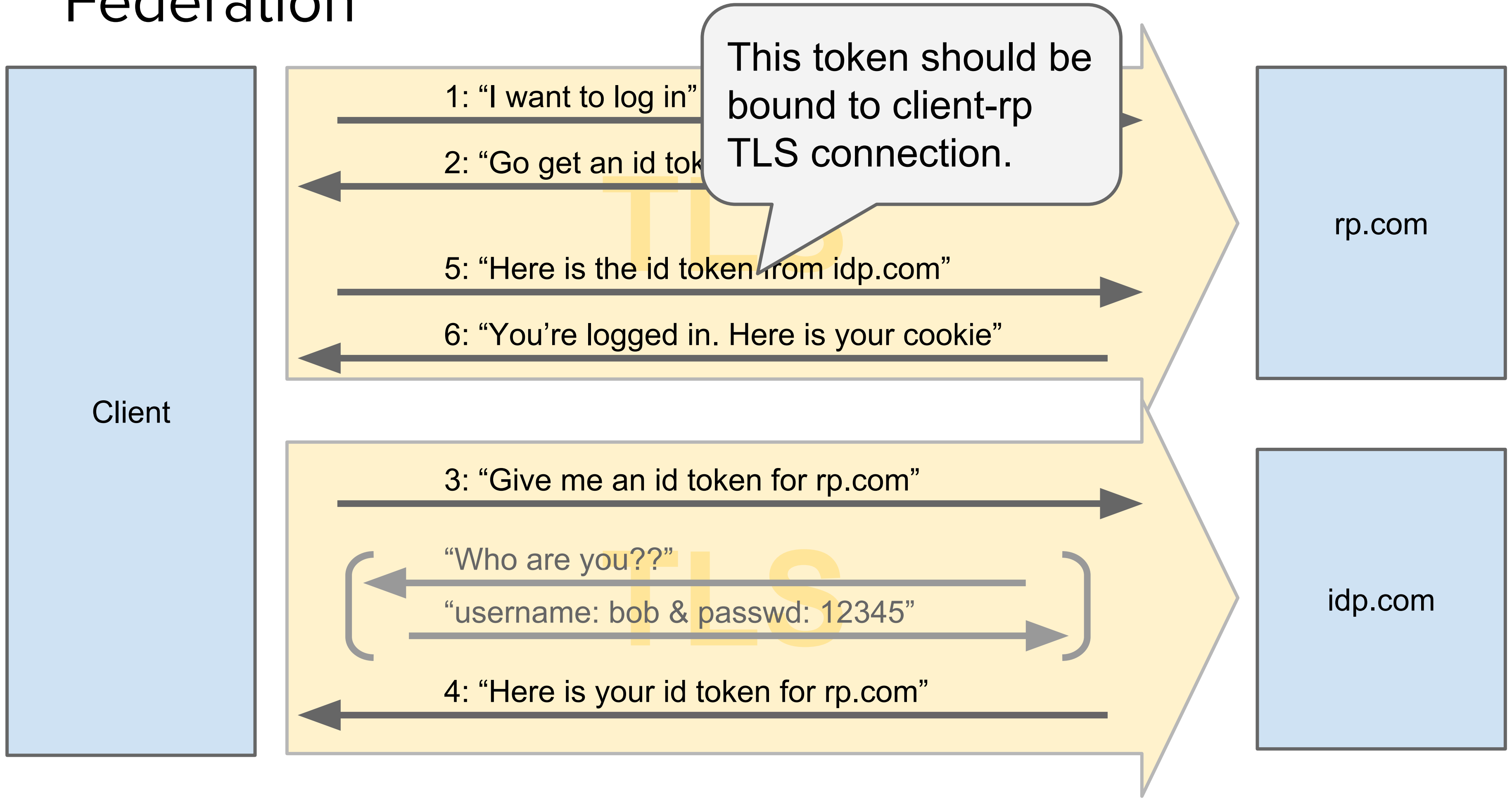
Federation


Client
rp.com
idp.com
TLS
1: “I want to log in”
2: “Go get an id token from idp.com”
TLS
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”

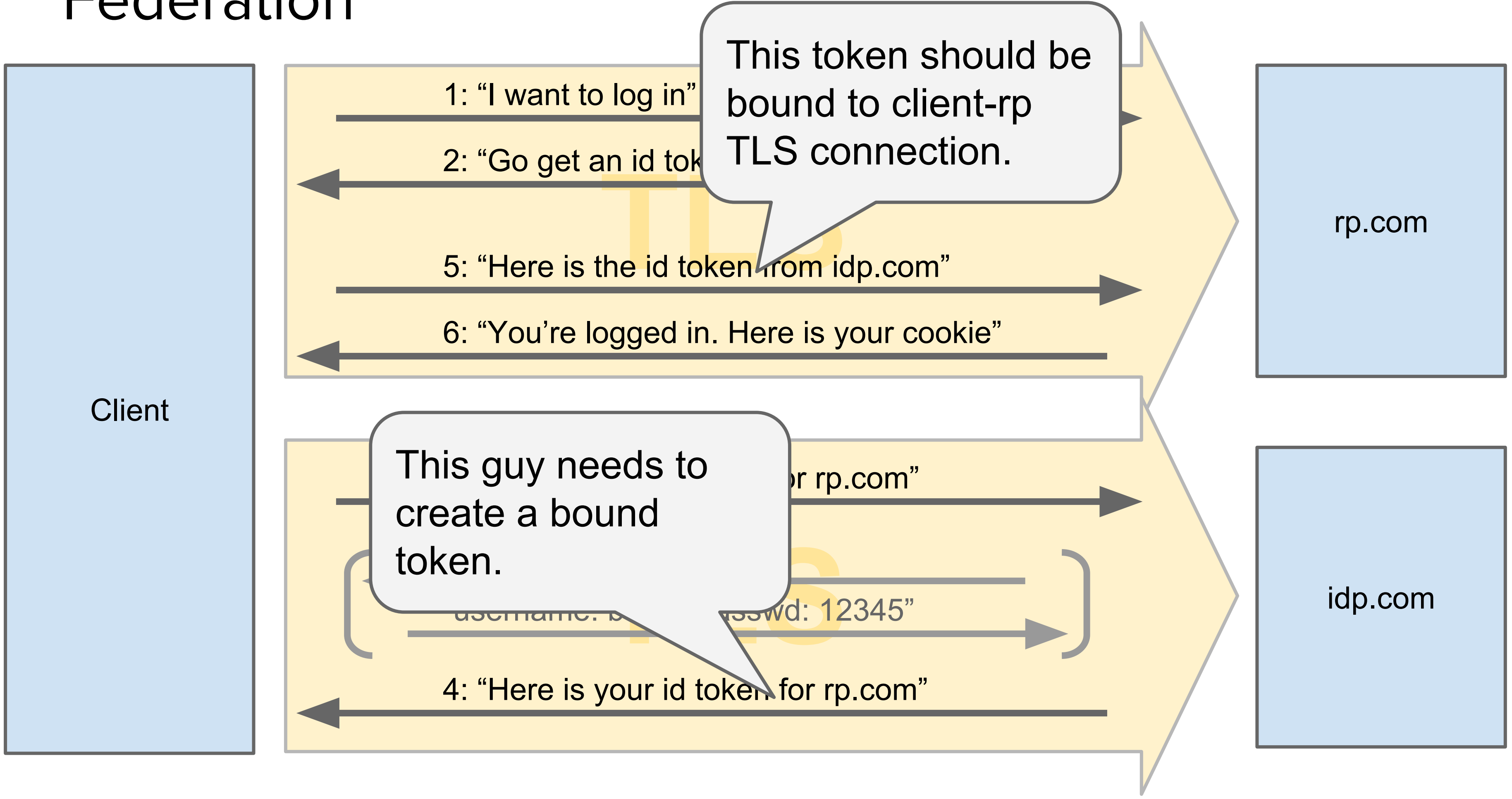
Federation


Client
rp.com
idp.com
TLS
1: "I want to log in"
2: "Go get an id token from idp.com"
5: "Here is the id token from idp.com"
6: "You're logged in. Here is your cookie"
TLS
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"

Federation

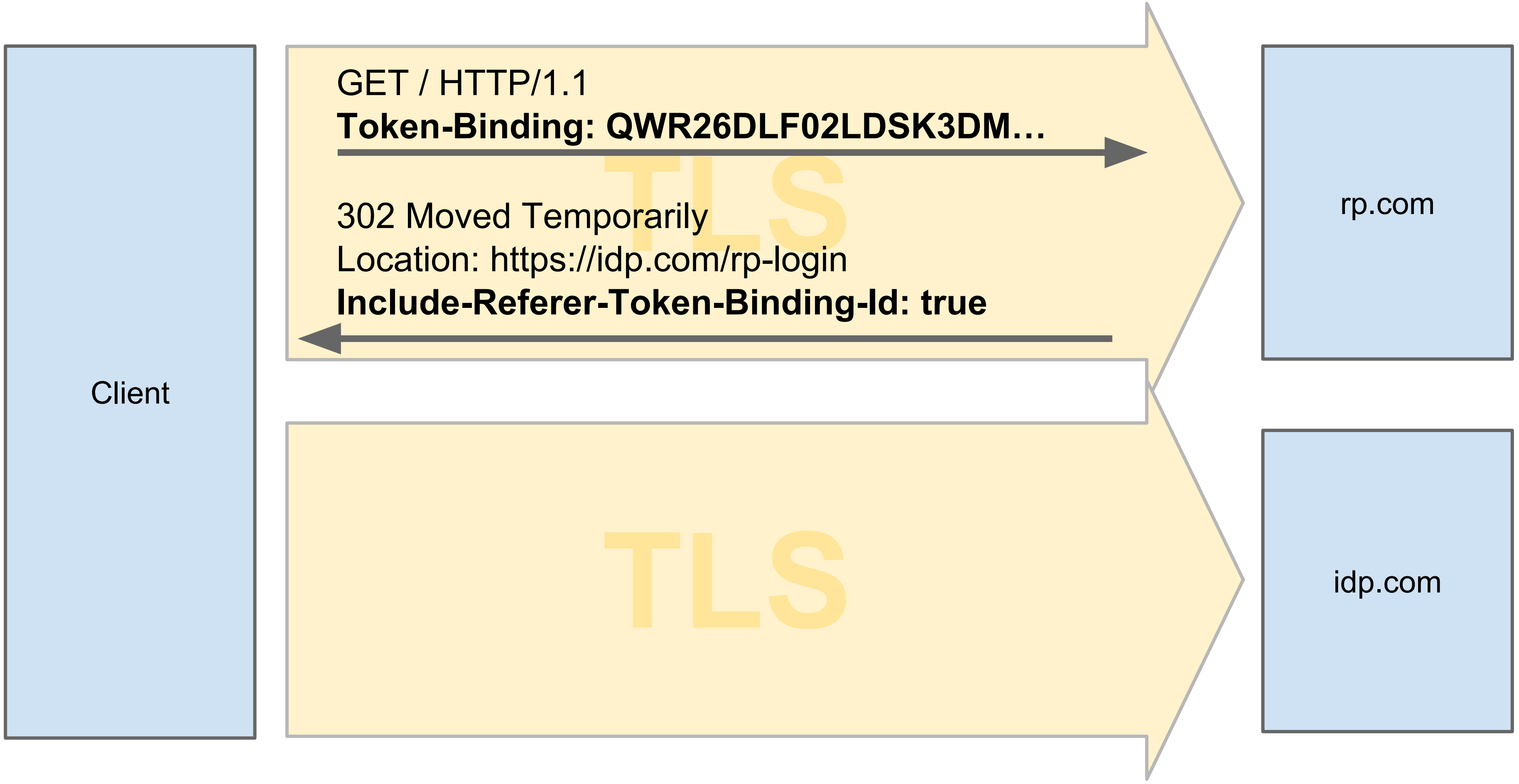

Client
1: "I want to log in"
2: "Go get an id tok
TLS
5: "Here is the id token from idp.com"
6: "You're logged in. Here is your cookie"
This token should be bound to client-rp TLS connection.
rp.com
3: "Give me an id token for rp.com"
"Who are you??"
"username: bob & passwd: 12345"
TLS
4: "Here is your id token for rp.com"
idp.com

Federation



Client
rp.com
idp.com
TLS
TLS
1: “I want to log in”
2: “Go get an id tok
5: “Here is the id token from idp.com”
6: “You’re logged in. Here is your cookie”
r rp.com”
username: b
sswd: 12345”
4: “Here is your id token for rp.com”
This token should be bound to client-rp TLS connection.
This guy needs to create a bound token.

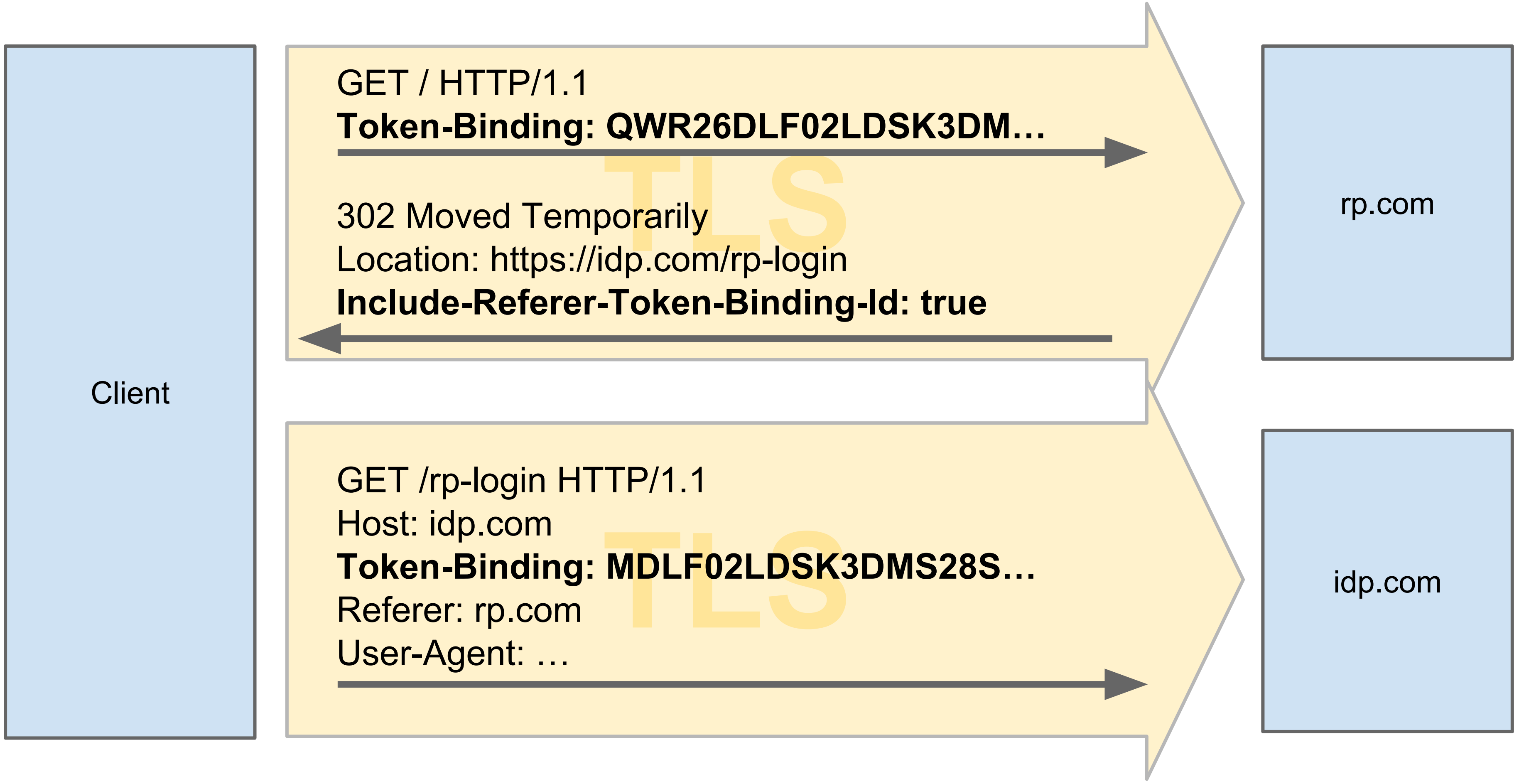
Federation with HTTP Redirects



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

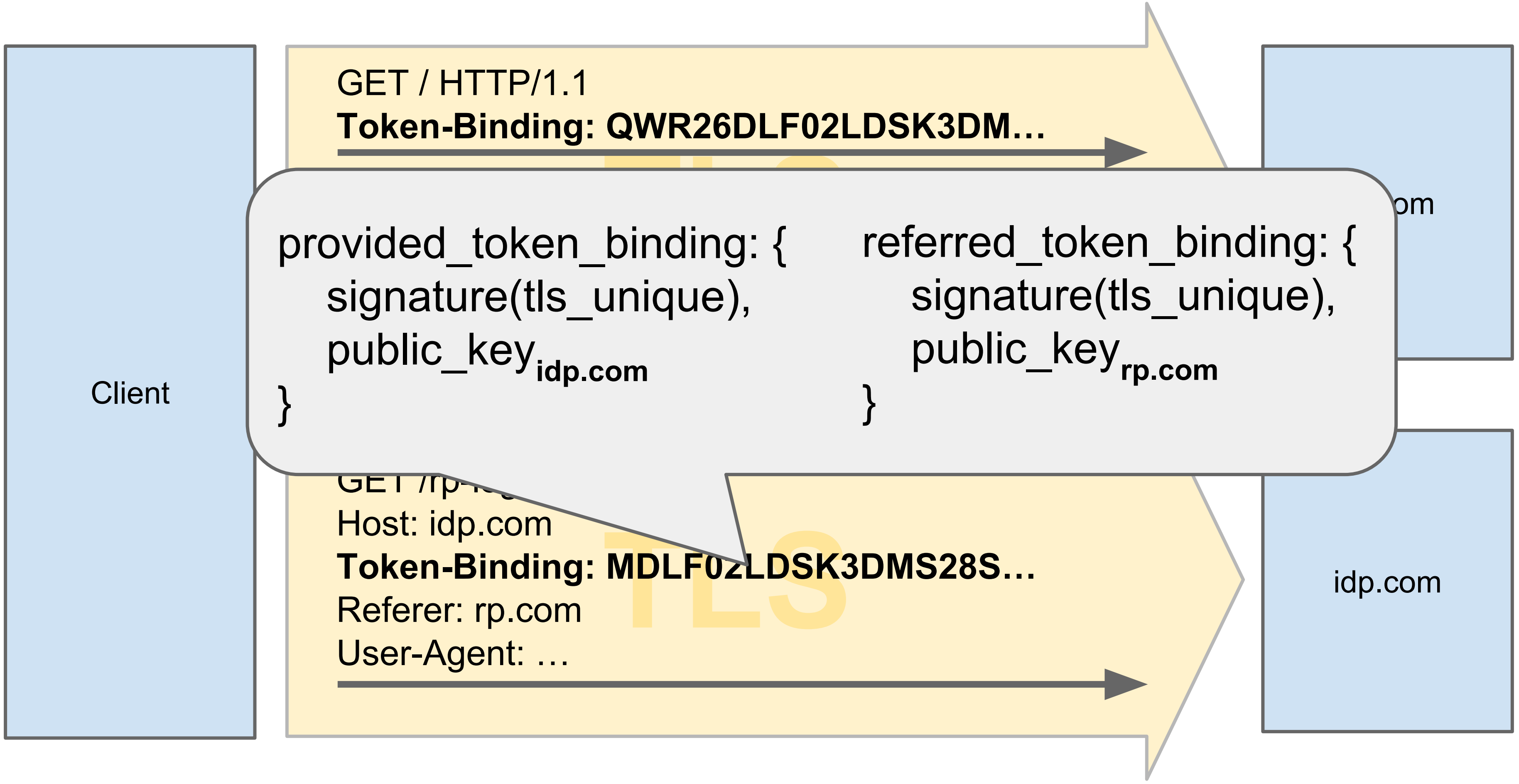


```
GET / HTTP/1.1
Token-Binding: QWR26DLF02LDSK3DM…
```

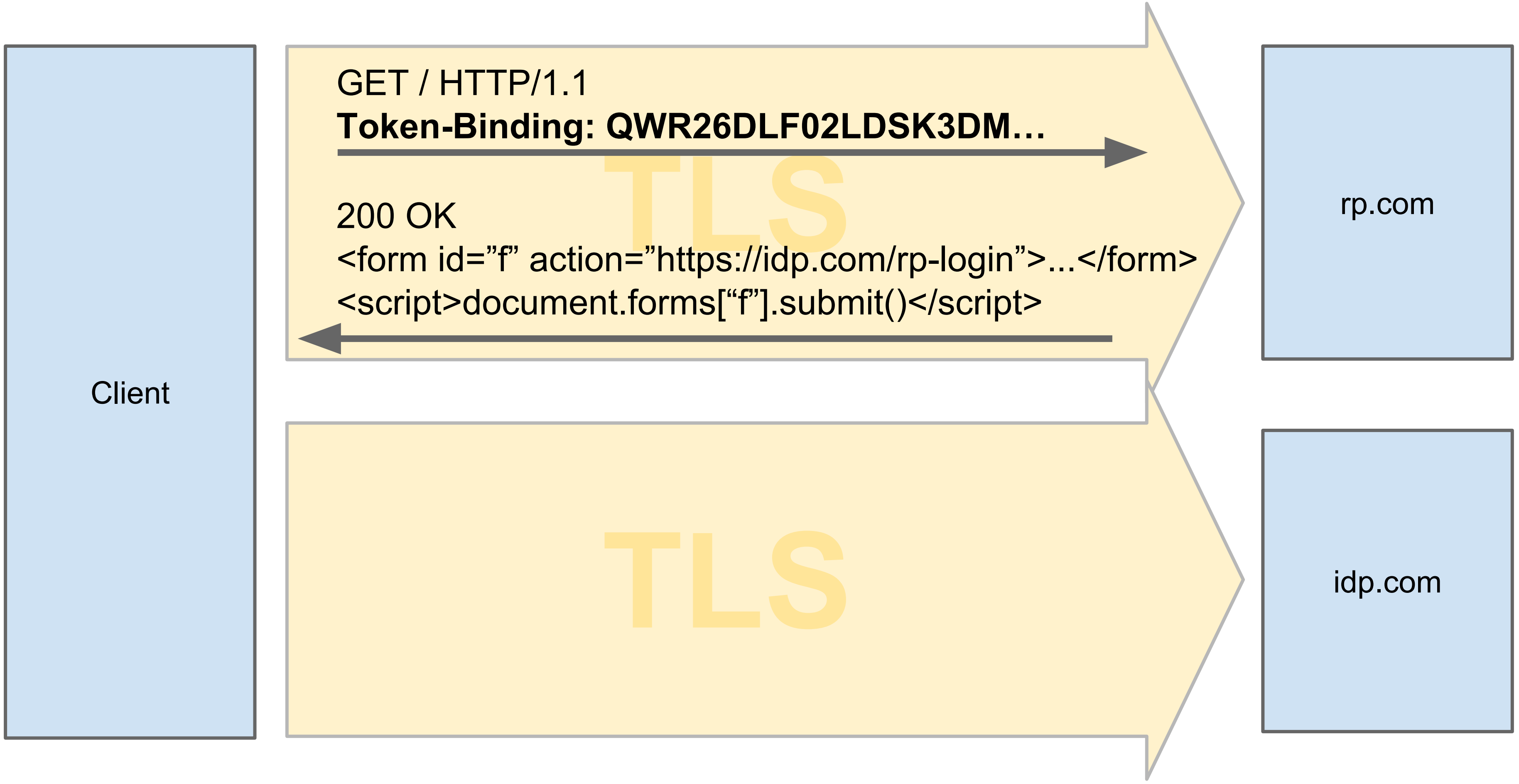
```
302 Moved Temporarily
Location: https://idp.com/rp-login
Include-Referer-Token-Binding-Id: true
```

```
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: QWR26DLF02LDSK3DM…
TLS
om
Client
provided_token_binding: {
signature(tls_unique),
public_key$_{\text{idp.com}}$
}
referred_token_binding: {
signature(tls_unique),
public_key$_{\text{rp.com}}$
}
GET /rp-lo
Host: idp.com
Token-Binding: MDLF02LDSK3DMS28S…
Referer: rp.com
User-Agent: …
TLS
idp.com

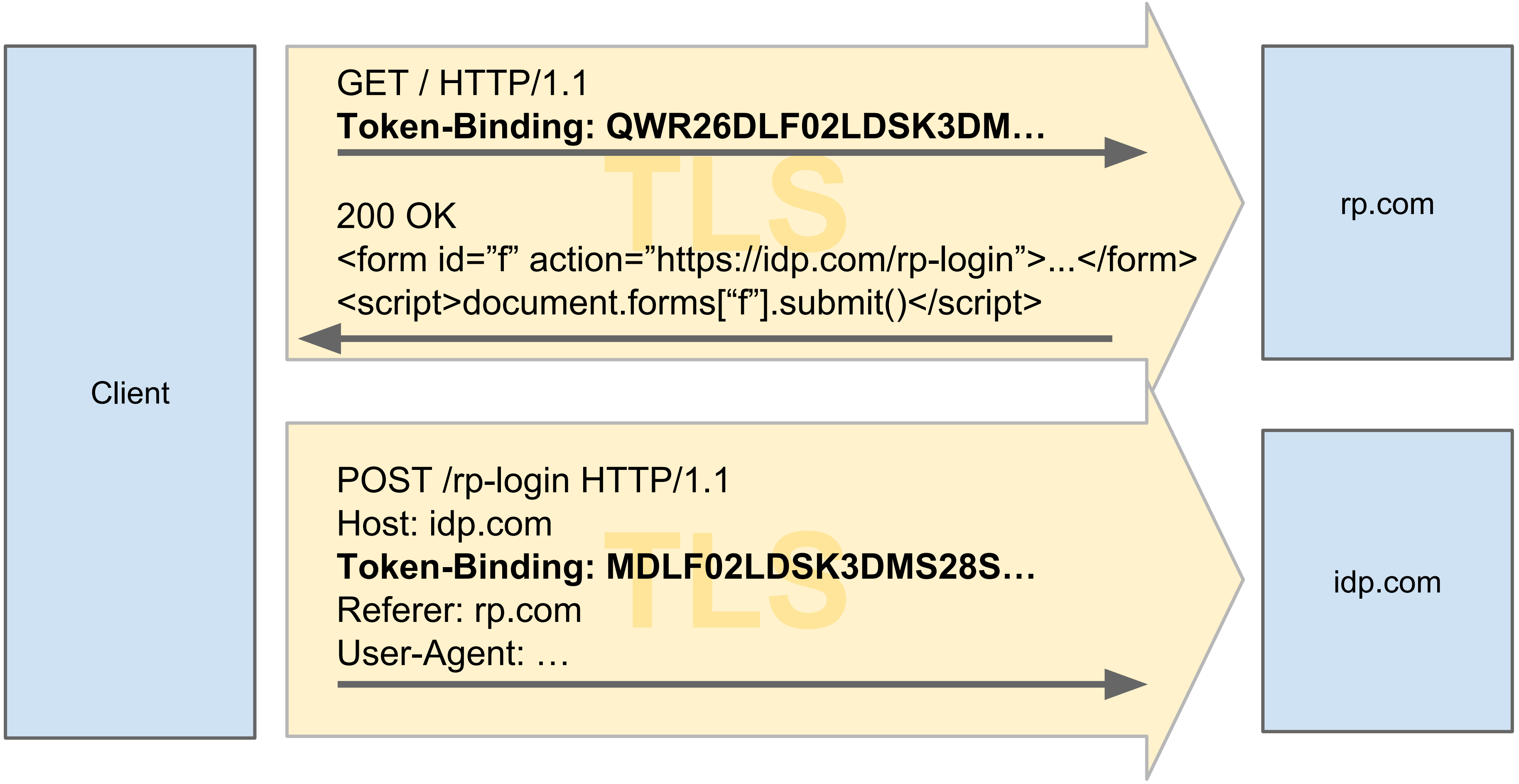
Federation with POSTs



GET / HTTP/1.1
Token-Binding: QWR26DLF02LDSK3DM…

200 OK
<form id=”f” action=”https://idp.com/rp-login”>...</form>
<script>document.forms[“f”].submit()</script>

Federation with POSTs

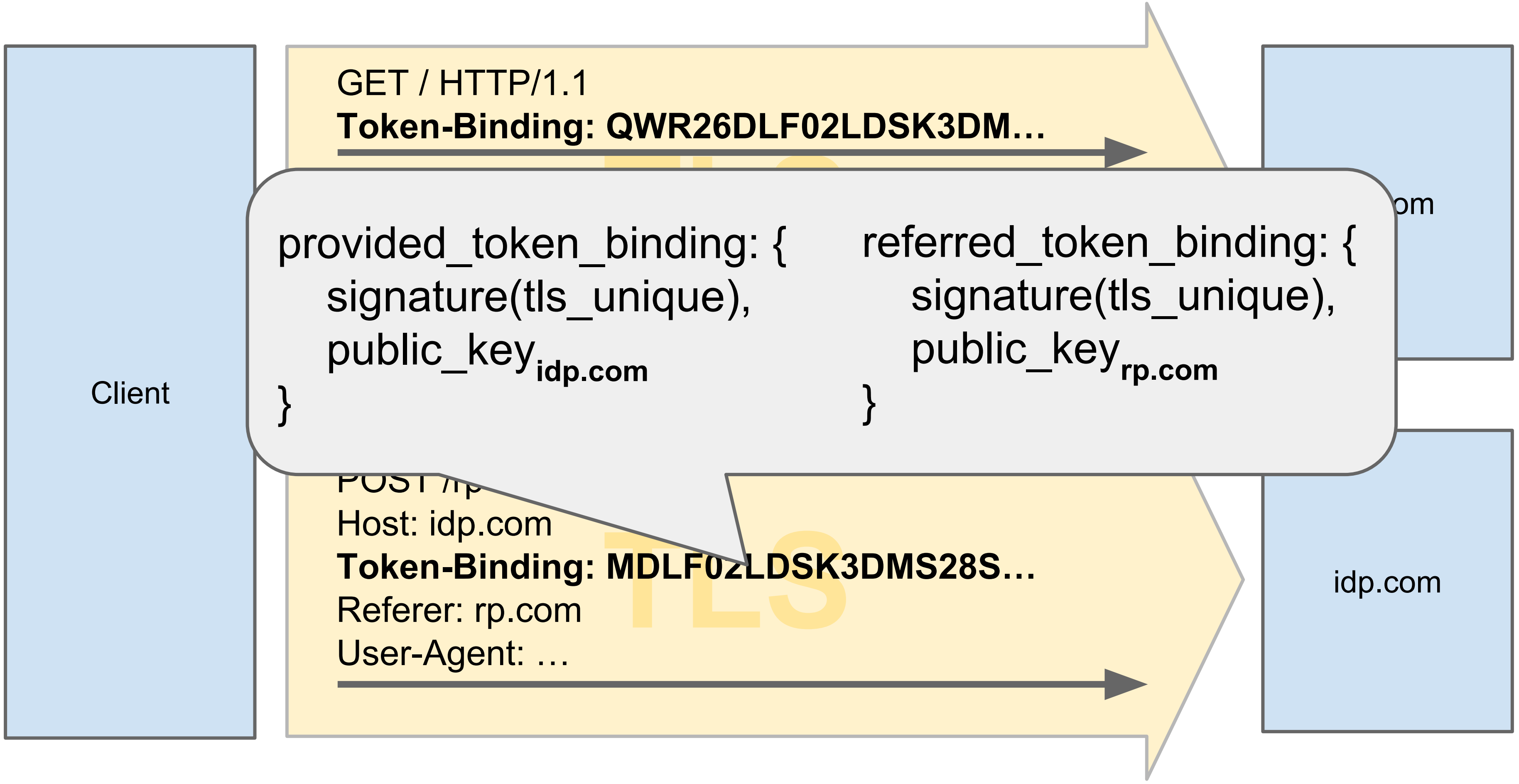


```
GET / HTTP/1.1
Token-Binding: QWR26DLF02LDSK3DM…
```

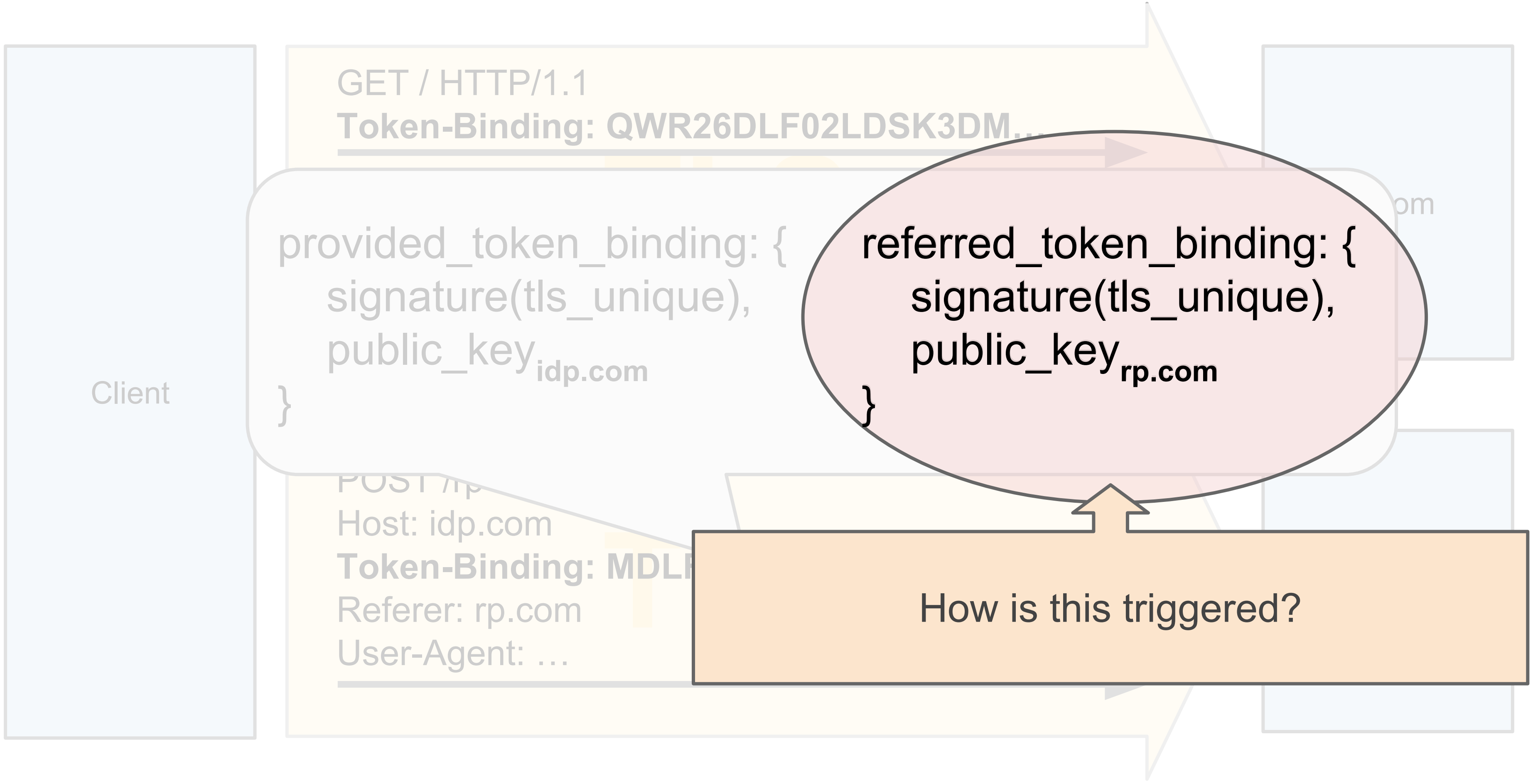
```
200 OK
<form id=”f” action=”https://idp.com/rp-login”>...</form>
<script>document.forms[“f”].submit()</script>
```

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

Federation with POSTs



Federation with POSTs



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

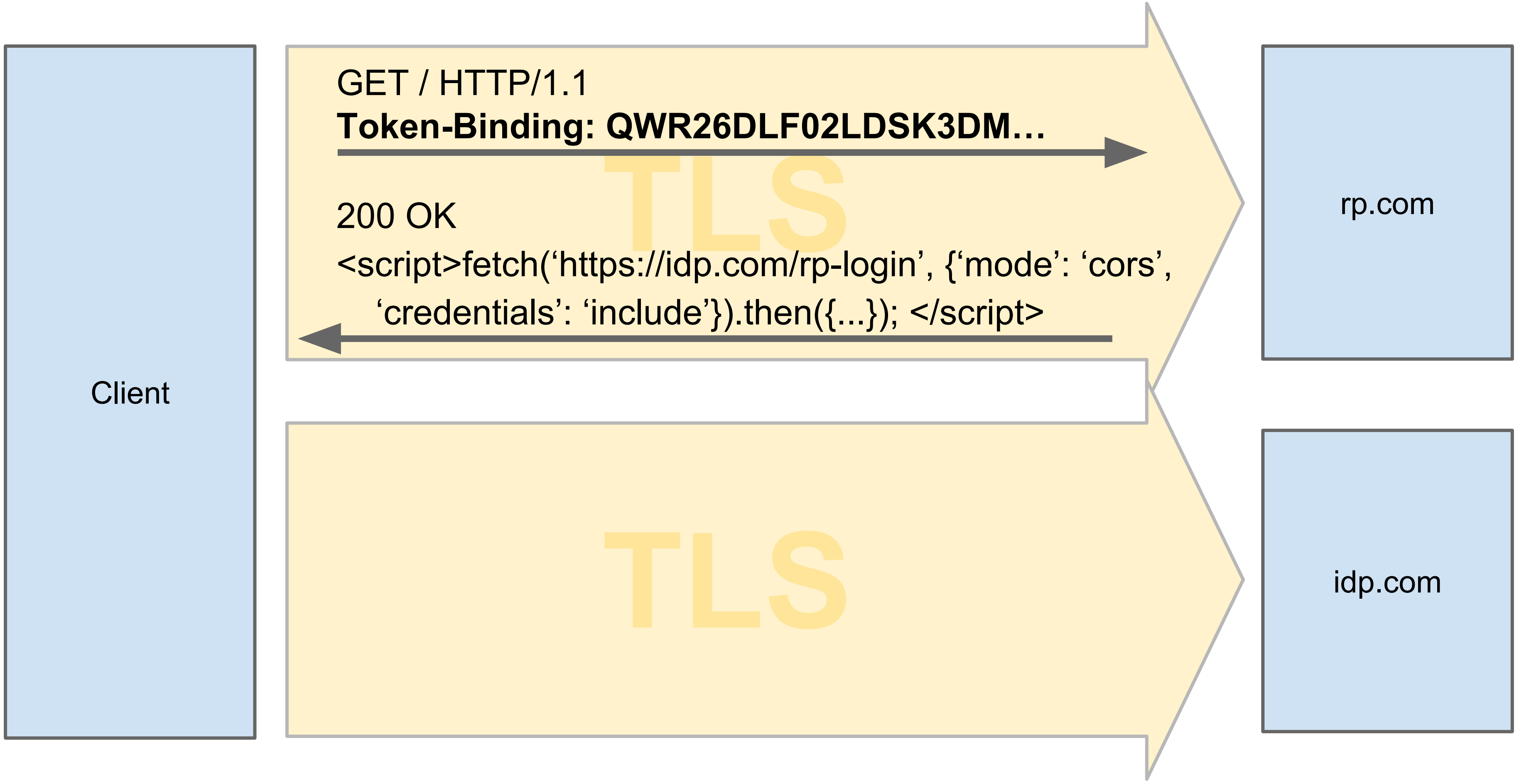
```
provided_token_binding: {
    signature(tls_unique),
    public_key_idp.com
}
```

```
POST /rp
Host: idp.com
Token-Binding: MDLF
Referer: rp.com
User-Agent: …
```

```
referred_token_binding: {
    signature(tls_unique),
    public_key_rp.com
}
```

How is this triggered?

Federation with XHRs

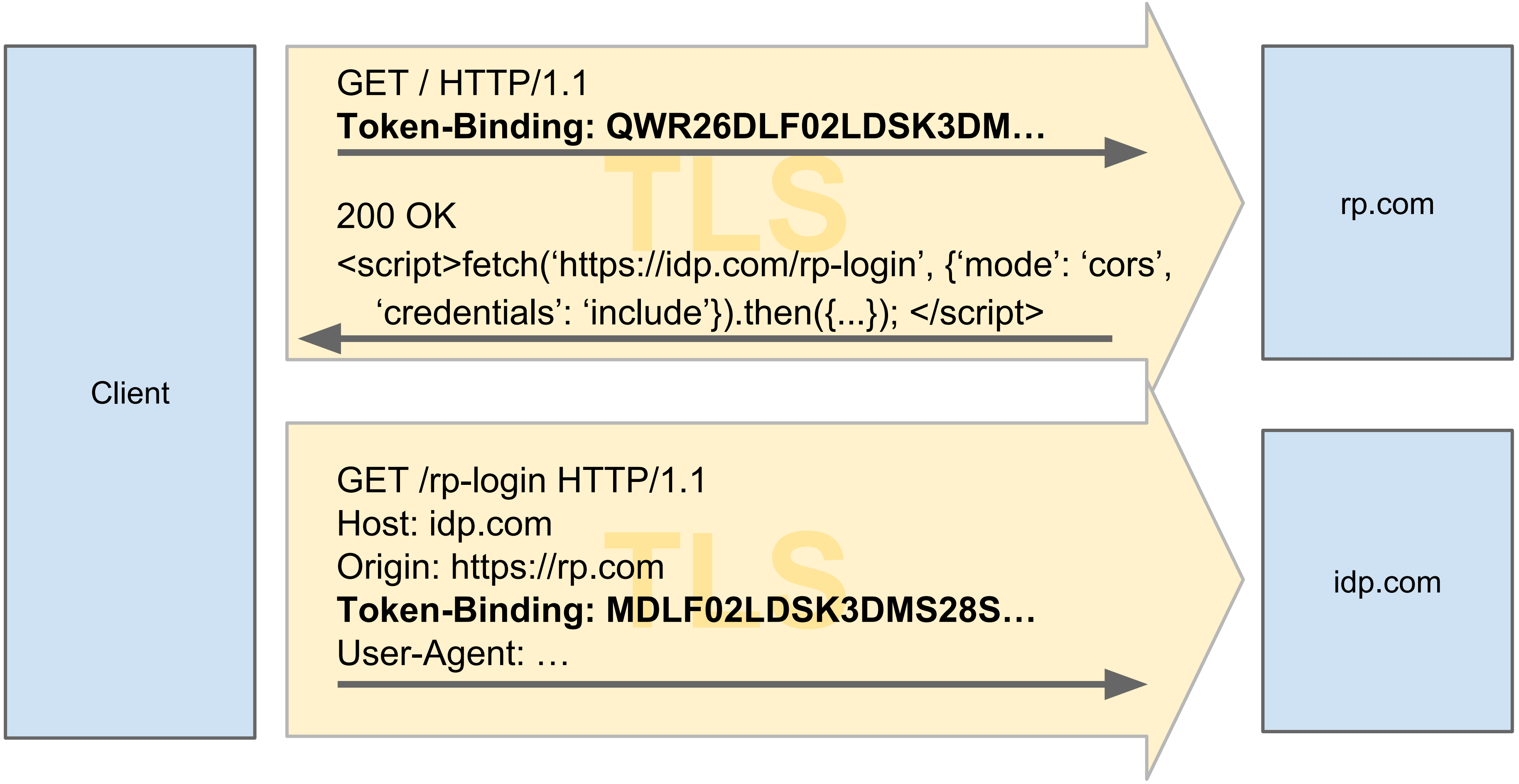


GET / HTTP/1.1
Token-Binding: QWR26DLF02LDSK3DM…

200 OK

```
<script>fetch('https://idp.com/rp-login', {'mode': 'cors',
    'credentials': 'include'}).then({...}); </script>
```

Federation with XHRs

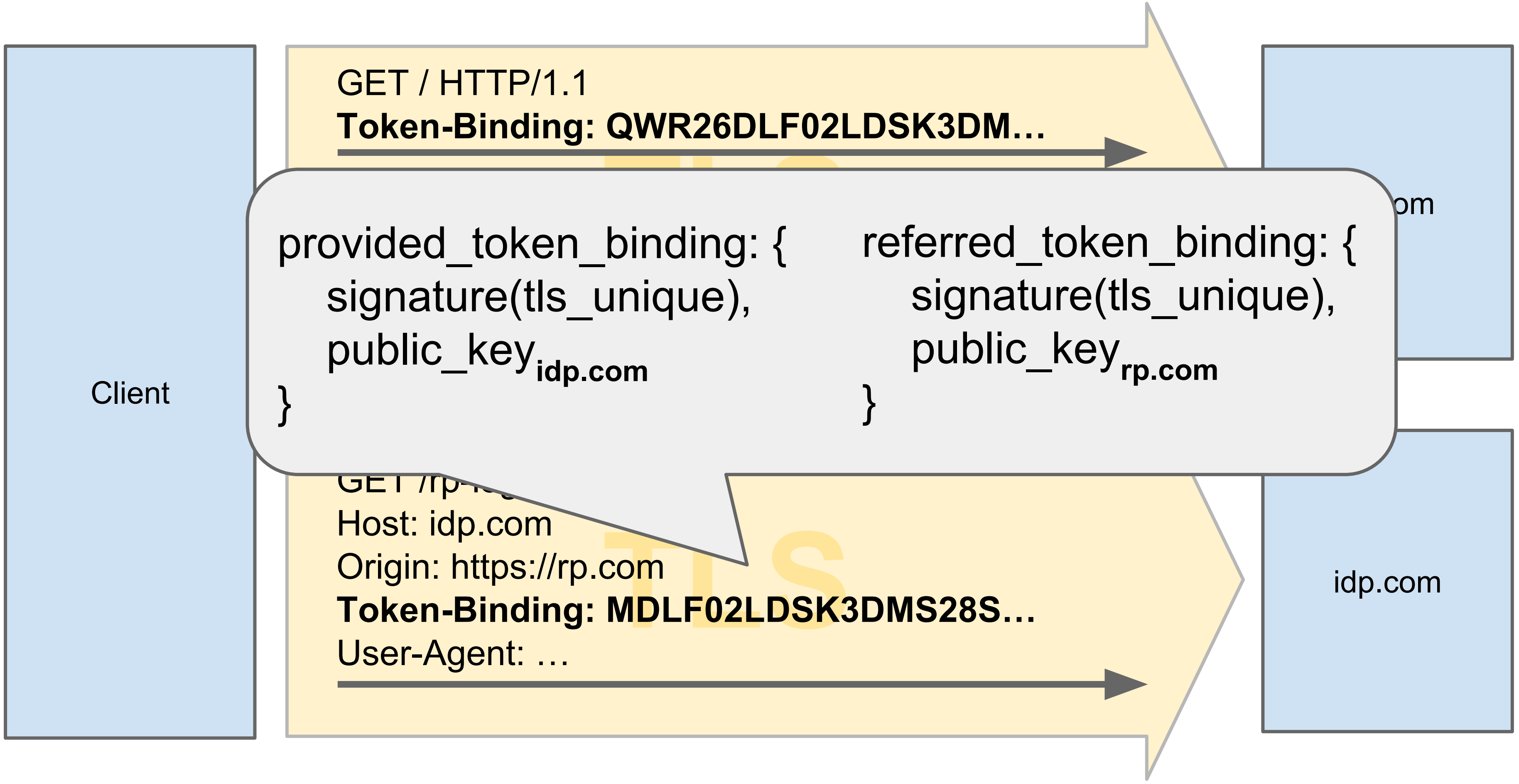


GET / HTTP/1.1
Token-Binding: QWR26DLF02LDSK3DM…

200 OK
<script>fetch('https://idp.com/rp-login', {'mode': 'cors', 'credentials': 'include'}).then({...}); </script>

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

Federation with XHRs



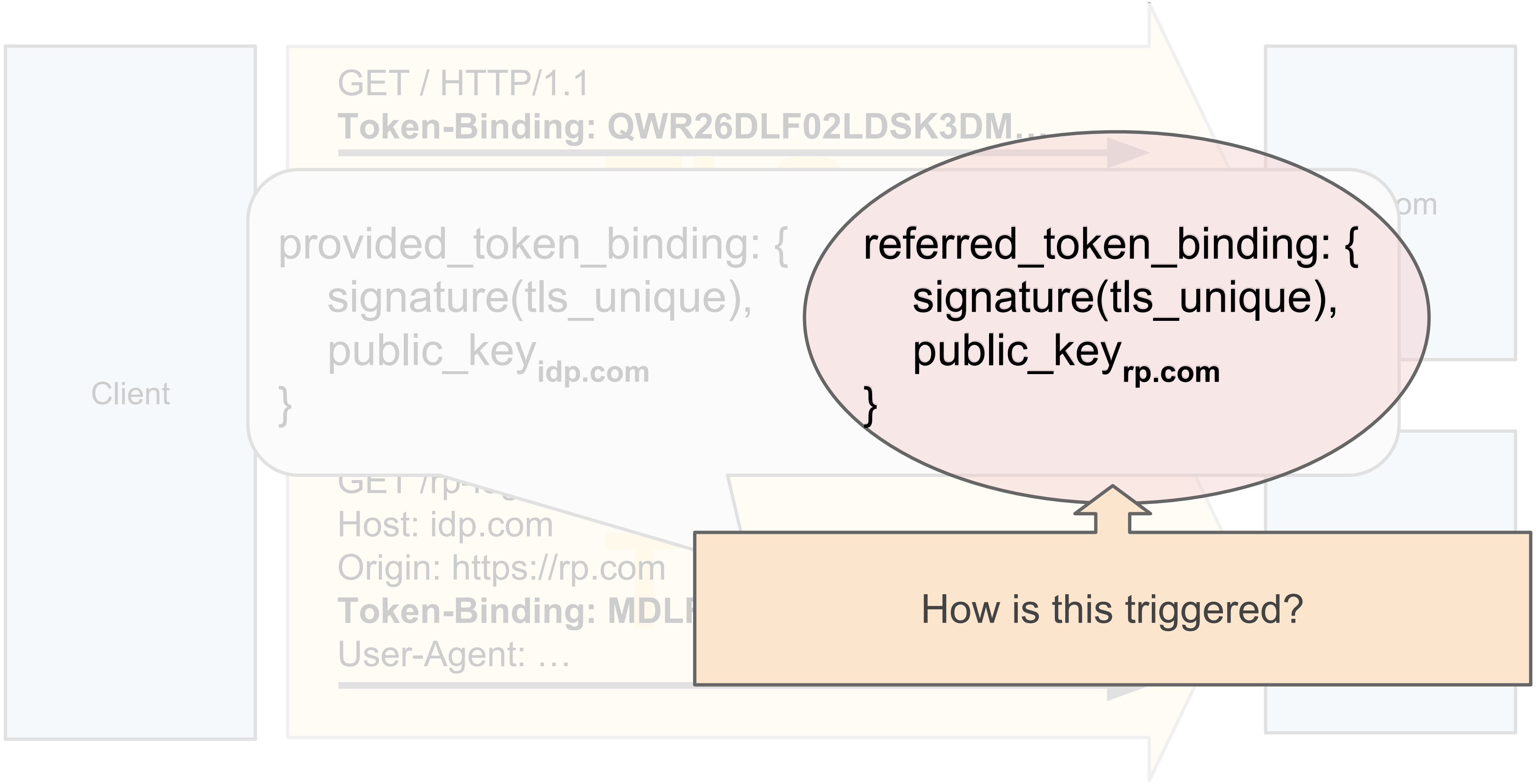
GET / HTTP/1.1
Token-Binding: QWR26DLF02LDSK3DM…

provided_token_binding: {
    signature(tls_unique),
    public_key$_{\text{idp.com}}$
}

referred_token_binding: {
    signature(tls_unique),
    public_key$_{\text{rp.com}}$
}

GET /rp-lo
Host: idp.com
Origin: https://rp.com
Token-Binding: MDLF02LDSK3DMS28S…
User-Agent: …

Federation with XHRs



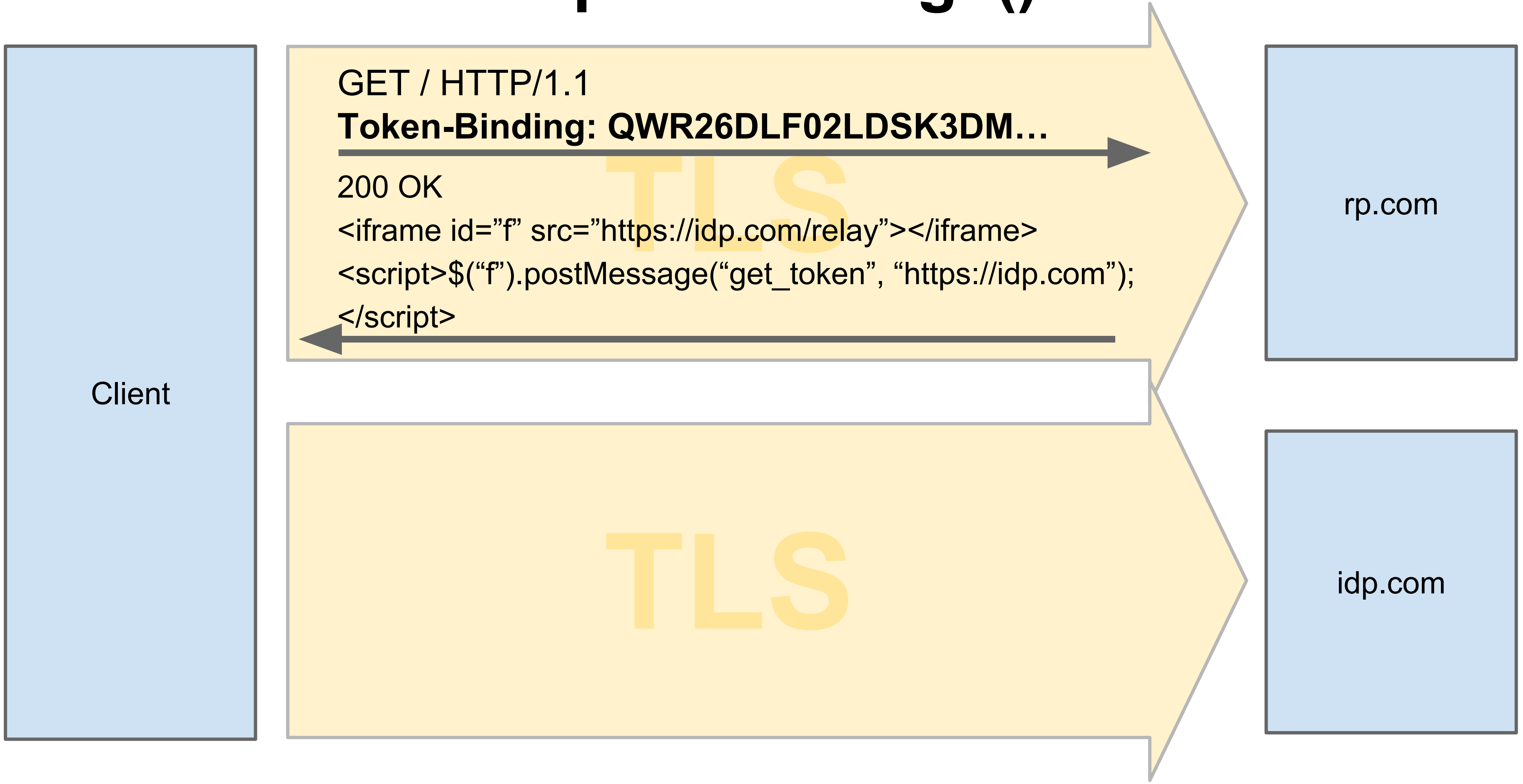
```
GET / HTTP/1.1
Token-Binding: QWR26DLF02LDSK3DM…
```

```
provided_token_binding: {
    signature(tls_unique),
    public_key_idp.com
}
```

```
referred_token_binding: {
    signature(tls_unique),
    public_key_rp.com
}
```

```
GET /rp-lo
Host: idp.com
Origin: https://rp.com
Token-Binding: MDL
User-Agent: …
```

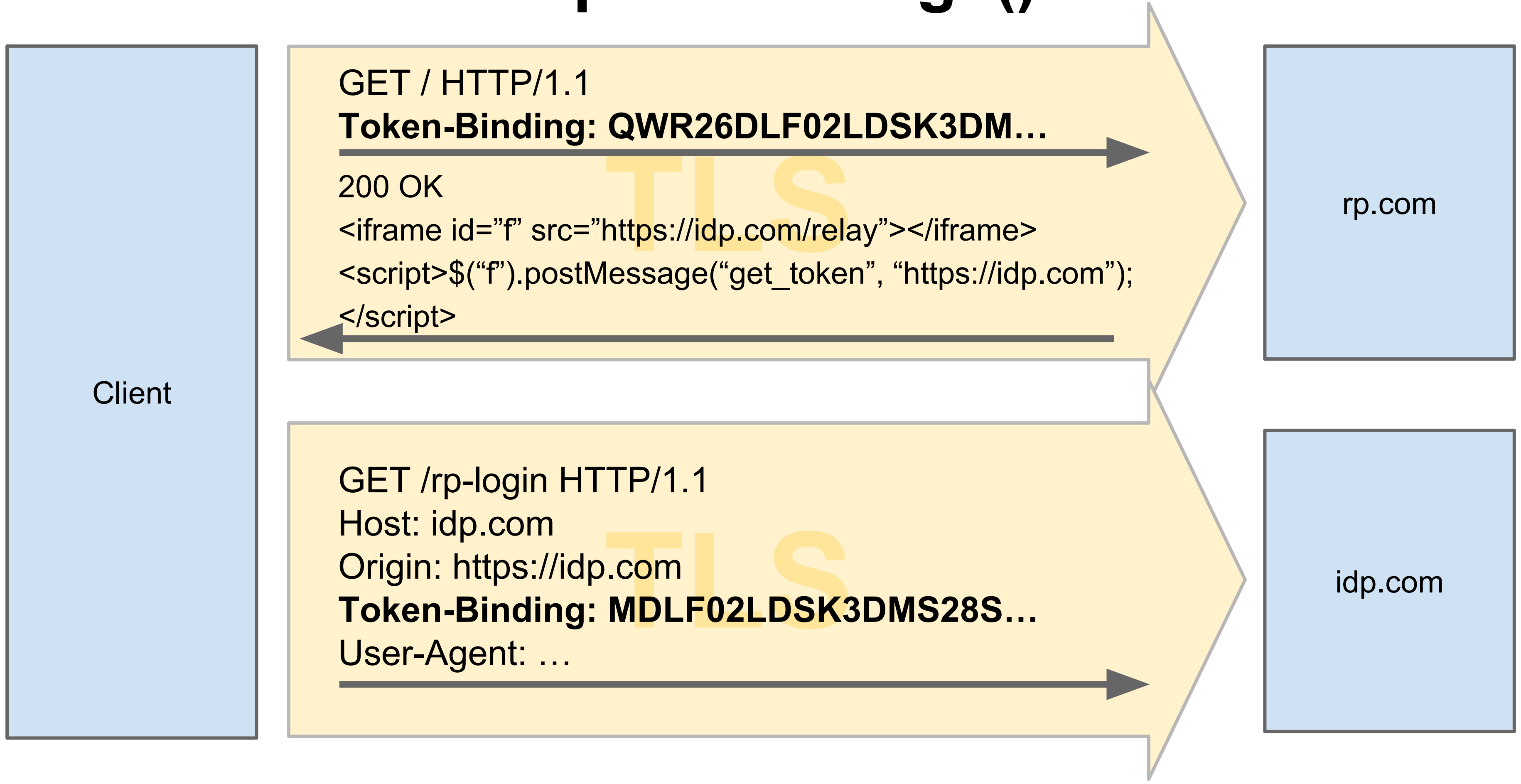
Federation with postMessage()



GET / HTTP/1.1
Token-Binding: QWR26DLF02LDSK3DM…

200 OK
<iframe id=”f” src=”https://idp.com/relay”></iframe>
<script>$(“f”).postMessage(“get_token”, “https://idp.com”);
</script>

Federation with postMessage()

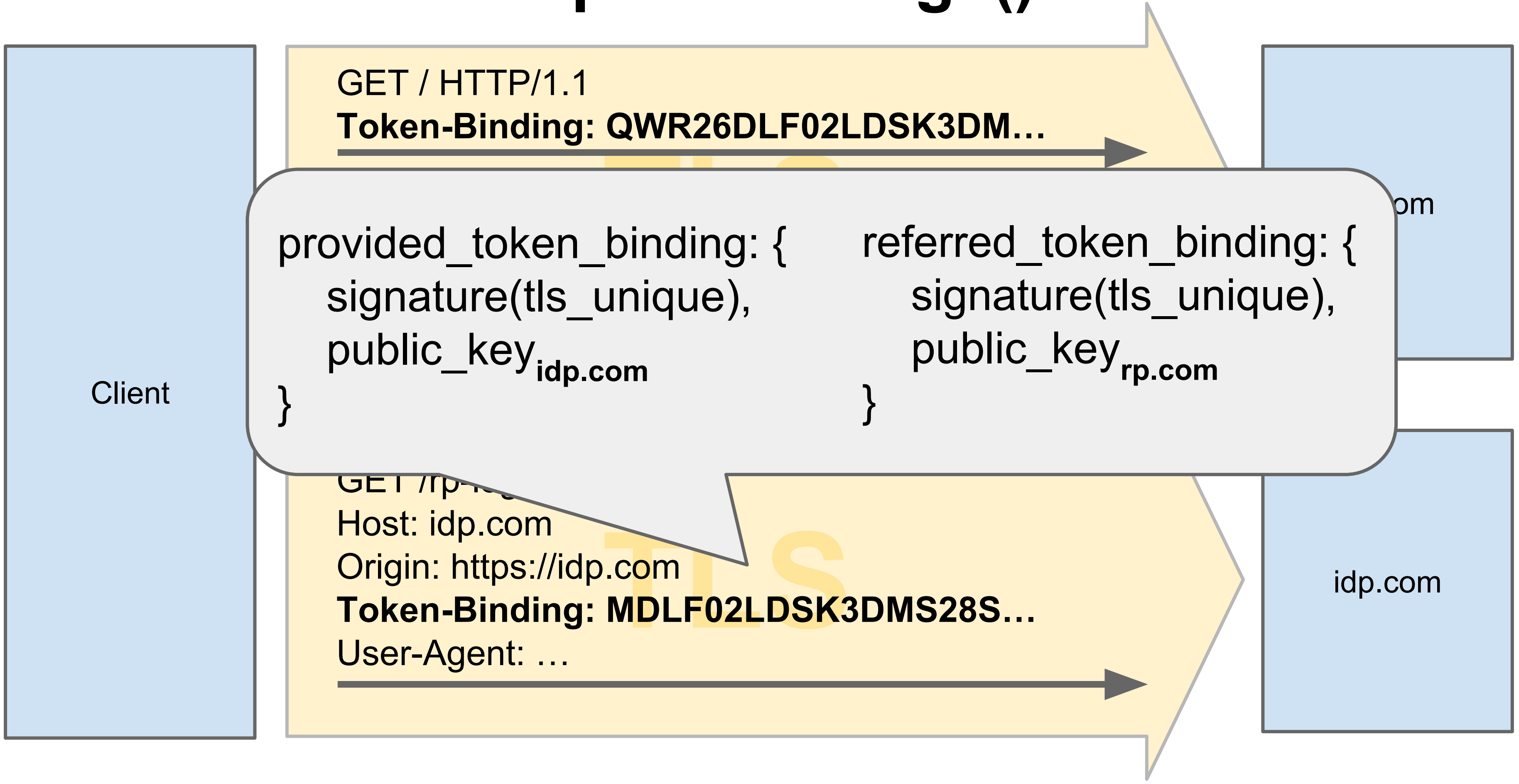


```
GET / HTTP/1.1
Token-Binding: QWR26DLF02LDSK3DM…
```

```
200 OK
<iframe id=”f” src=”https://idp.com/relay”></iframe>
<script>$(“f”).postMessage(“get_token”, “https://idp.com”);
</script>
```

```
GET /rp-login HTTP/1.1
Host: idp.com
Origin: https://idp.com
Token-Binding: MDLF02LDSK3DMS28S…
User-Agent: …
```

Federation with postMessage()



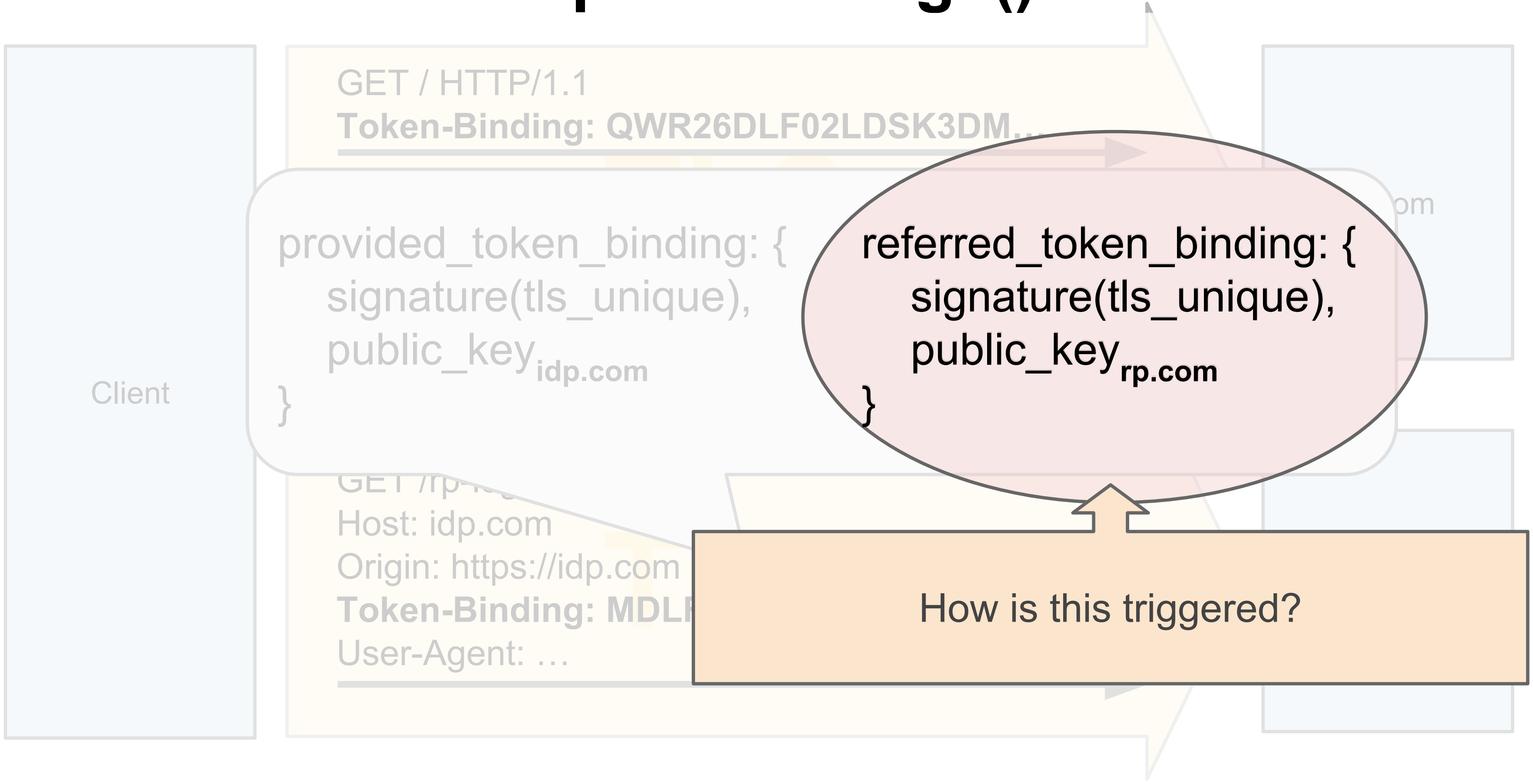
```
GET / HTTP/1.1
Token-Binding: QWR26DLF02LDSK3DM…
```

provided_token_binding: {
  signature(tls_unique),
  public_key$_{\text{idp.com}}$
}

referred_token_binding: {
  signature(tls_unique),
  public_key$_{\text{rp.com}}$
}

```
GET /rp-lo
Host: idp.com
Origin: https://idp.com
Token-Binding: MDLF02LDSK3DMS28S…
User-Agent: …
```

Federation with postMessage()



How to Trigger Referred Token Bindings?

Redirect-based

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

POST-based

```
<form id="f" action="https://idp.com/rp-login"
             refererTokenBinding="true">...</form>
<script><script>document.forms["f"].submit()</script>
```

XHR-based

```
fetch('https://idp.com/rp-login',
      {'mode': 'cors', 'credentials':'include',
      'refererTokenBinding': 'include'}).then({...});
```

postMessage()-based

```
???
```

Where to Specify Referred Token Bindings?

Redirect-based

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

IETF TokBind

POST-based

```
<form id=”f” action=”https://idp.com/rp-login”
             refererTokenBinding=”true”>...</form>
<script><script>document.forms[“f”].submit()</script>
```

XHR-based

```
fetch(‘https://idp.com’,
      {‘mode’: ‘cors’, ‘credentials’:‘include’,
      ‘refererTokenBinding’: ‘include’}).then({...});
```

postMessage()-based

```
                          ???
```

Should be discussed elsewhere.

Decisions to make

- Sec-Token-Binding vs. Token-Binding
- Extent of federation support spelled out in spec