The Art and Craft of Secrets

MICHAEL SWIETON

SWIETON@ATOMICOBJECT.COM

ATOMIC OBJECT


“CATCH ME IF YOU CAN” - DREAMWORKS, 2002


DO YOU TRUST ME?
“CATCH ME IF YOU CAN” - DREAMWORKS, 2002

ATOMIC OBJECT, MARCH 2016

HTTPS://ATOMICOBJECT.COM/CAREERS

ATOMIC OBJECT

THIS IS NOT A TALK ABOUT LOCKS AND CIPHERS.

THIS IS A TALK ABOUT BUYING EMBARRASSING THINGS ON THE INTERNET.

… IN SECRET.

THIS IS NOT A TALK ABOUT LOCKS AND CIPHERS.

THIS IS A TALK ABOUT BUYING EMBARRASSING THINGS ON THE INTERNET.

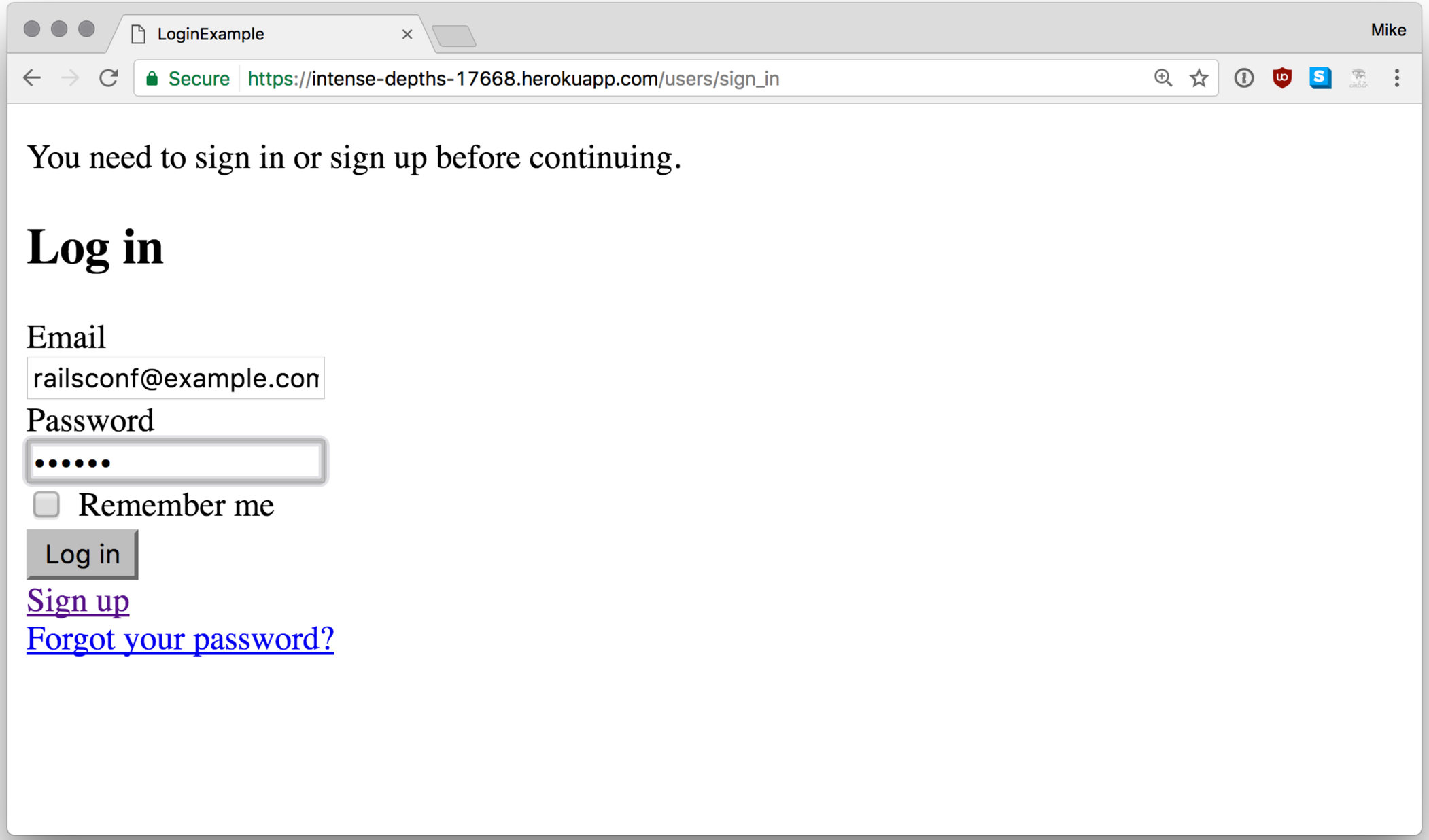
… IN SECRET.

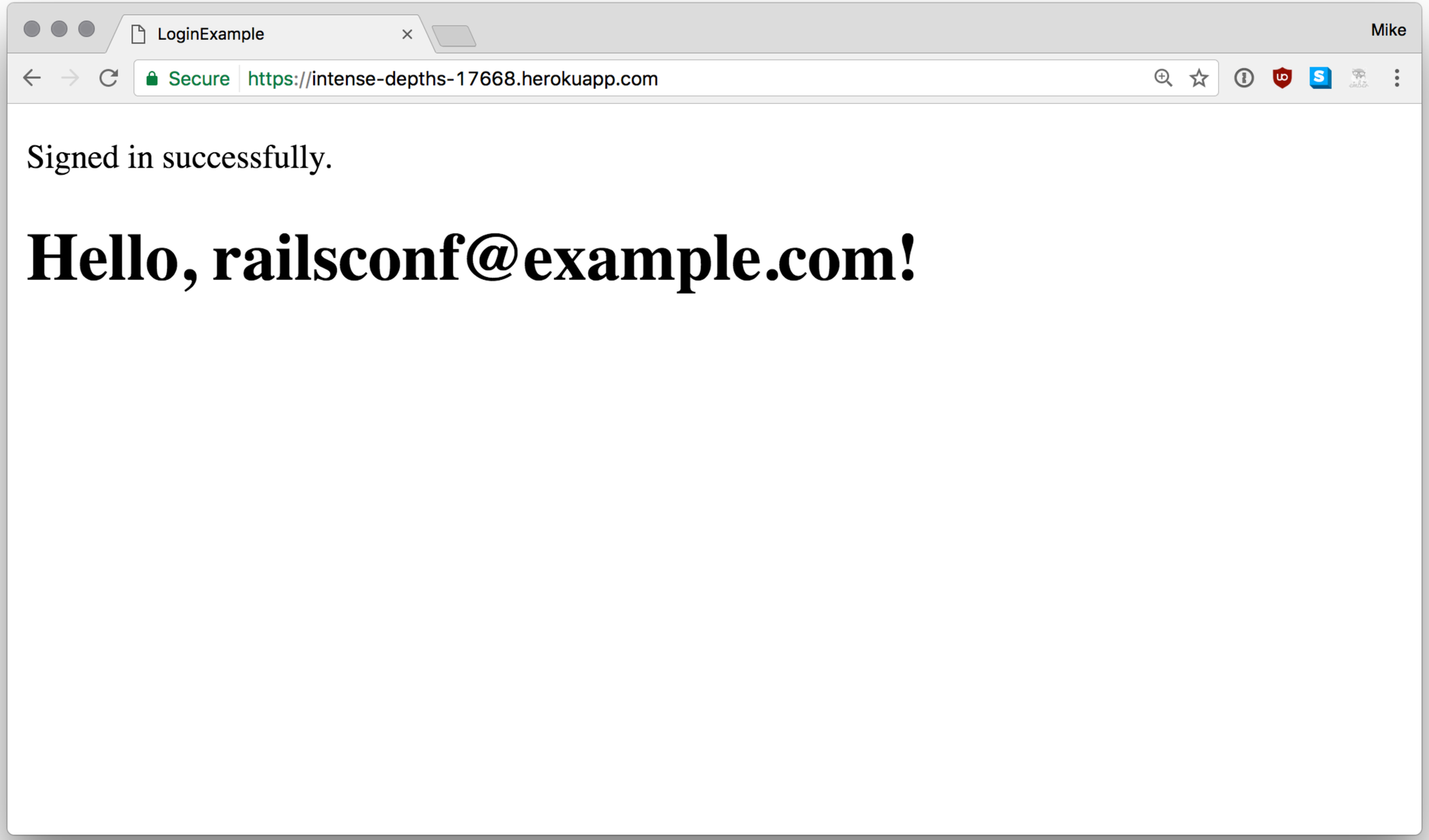
THIS IS NOT A TALK ABOUT LOCKS AND CIPHERS.

THIS IS A TALK ABOUT BUYING EMBARRASSING THINGS ON THE INTERNET.

… IN SECRET.

ATOMIC OBJECT


LoginExample
Mike
Secure
https://intense-depths-17668.herokuapp.com/users/sign_in
You need to sign in or sign up before continuing.
Log in
Email
railsconf@example.con
Password
Remember me
Log in
Sign up
Forgot your password?


LoginExample
Mike
Secure
https://intense-depths-17668.herokuapp.com
Signed in successfully.
Hello, railsconf@example.com!

DNS lookup for server

DNS lookup for server

Client connects to server via TCP

DNS lookup for server

Client connects to server via TCP

Establish secure connection

DNS lookup for server

Client connects to server via TCP

Establish secure connection

Send login credentials to server

DNS lookup for server

Client connects to server via TCP

Establish secure connection

Send login credentials to server

Server verifies login credentials

DNS lookup for server

Client connects to server via TCP

Establish secure connection

Send login credentials to server

Server verifies login credentials

Log user in

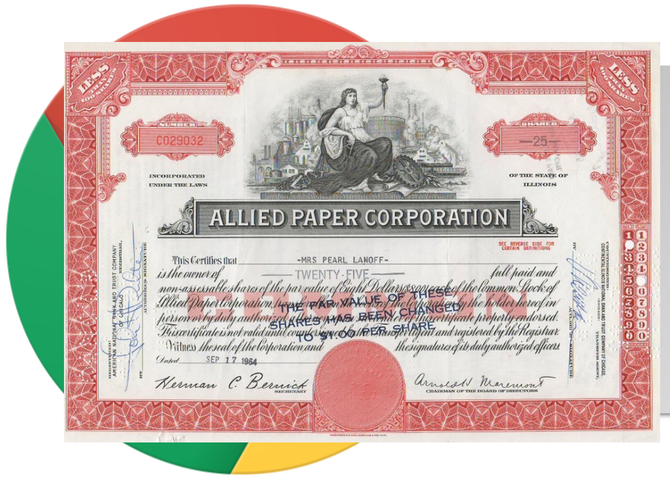
DNS lookup for server

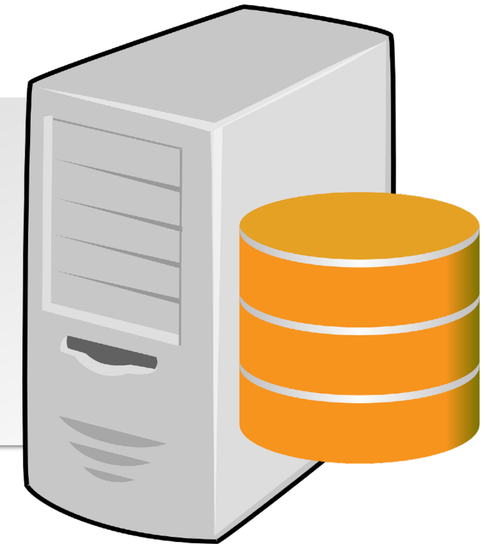
Client connects to server via TCP

DNS lookup for server

Client connects to server via TCP

Server sends certificate

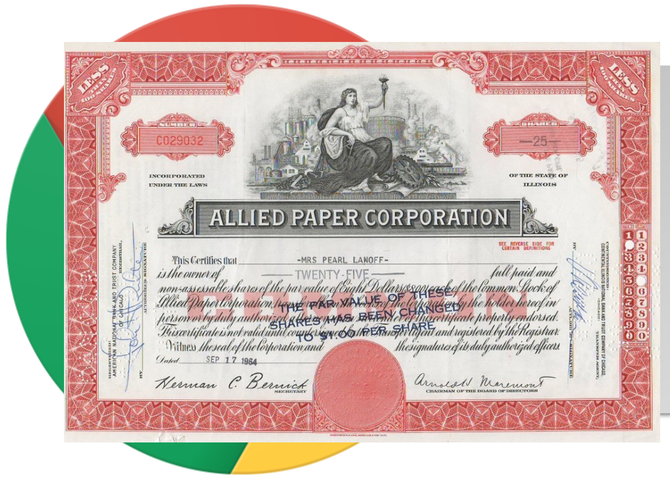




DNS lookup for server

Client connects to server via TCP

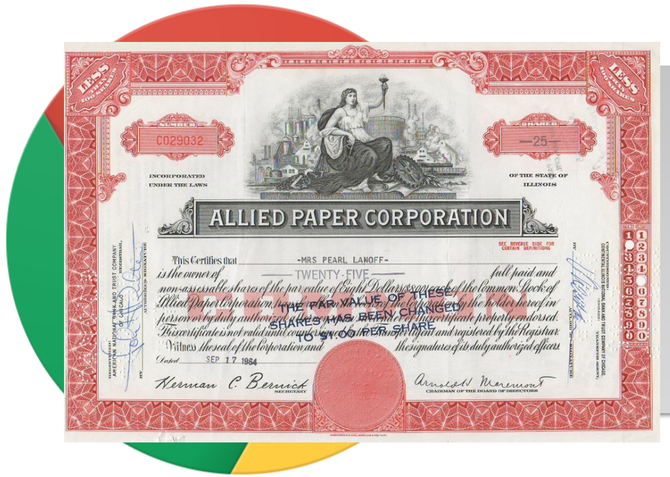
Server sends certificate

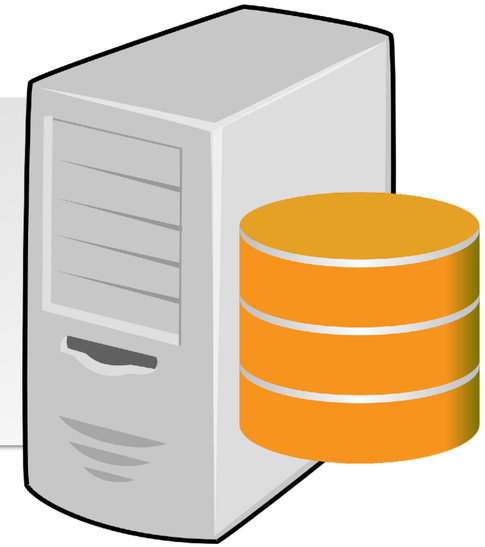


DNS lookup for server

Client connects to server via TCP

Server sends certificate





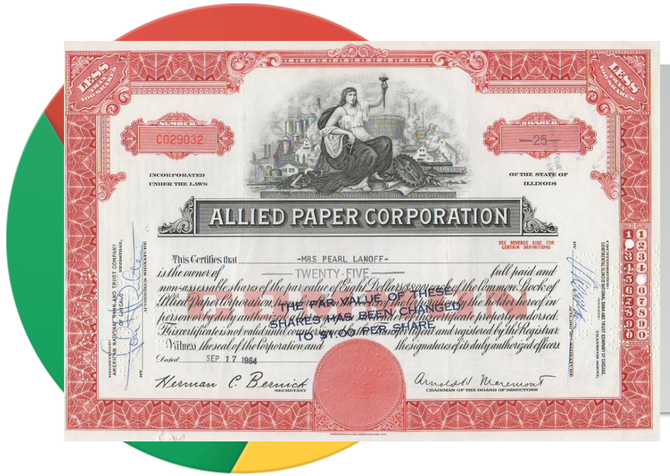
DNS lookup for server

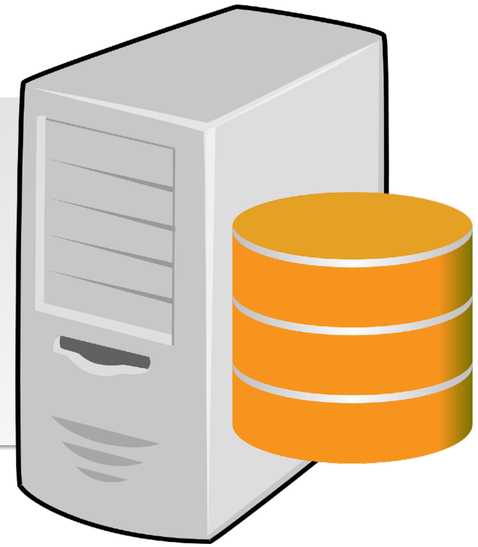
Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain





DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange

DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange

DNS lookup for server

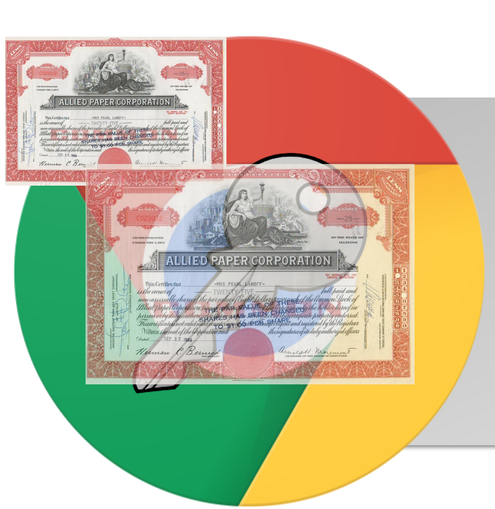
Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange



DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange

DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange

DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange

DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange

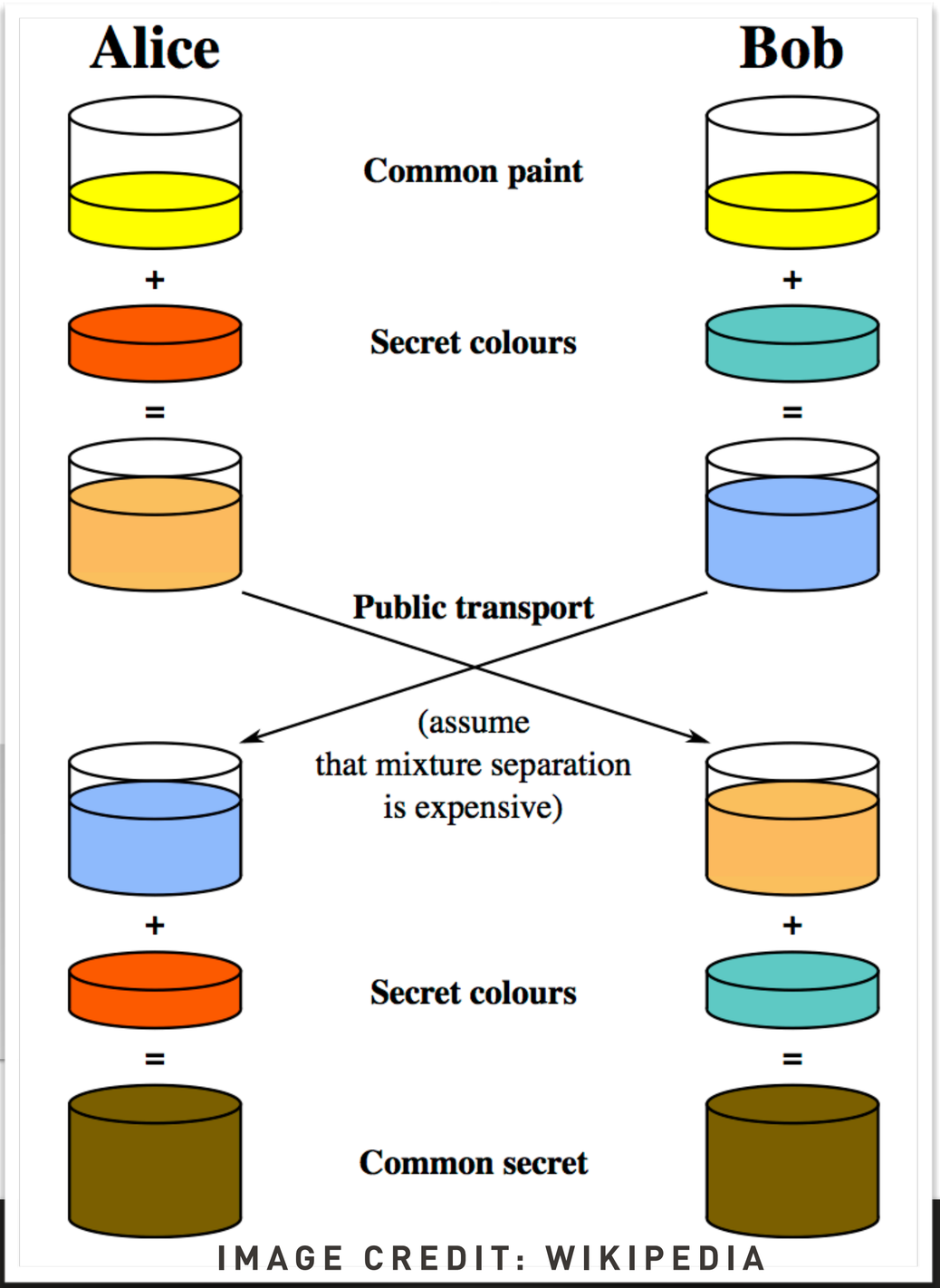


IMAGE CREDIT: WIKIPEDIA

DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange

DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange

DNS lookup for server

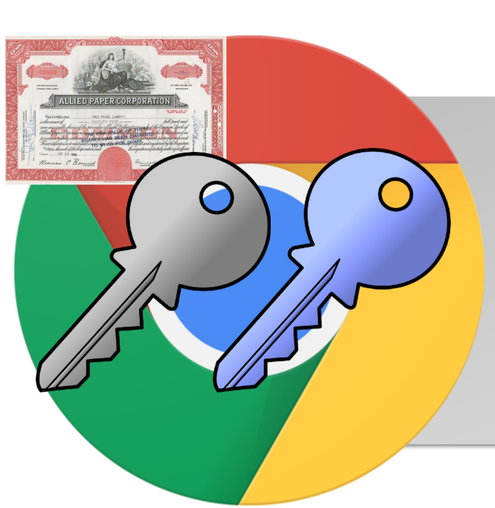
Client connects to server via TCP

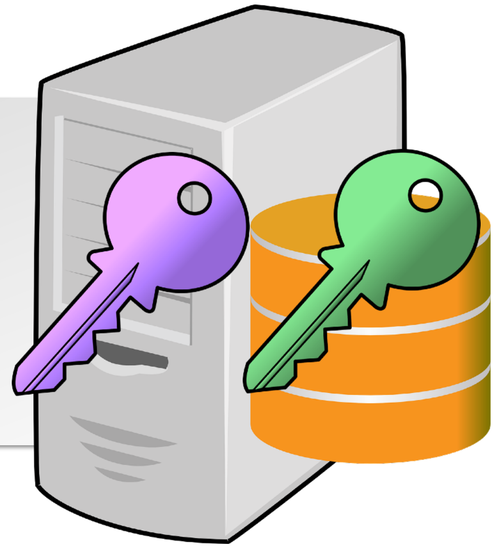
Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange





DNS lookup for server

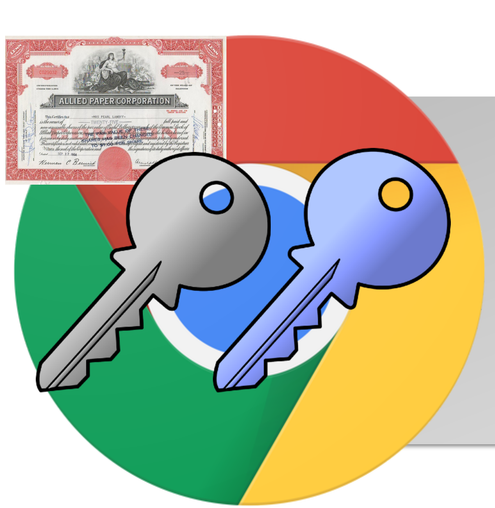
Client connects to server via TCP

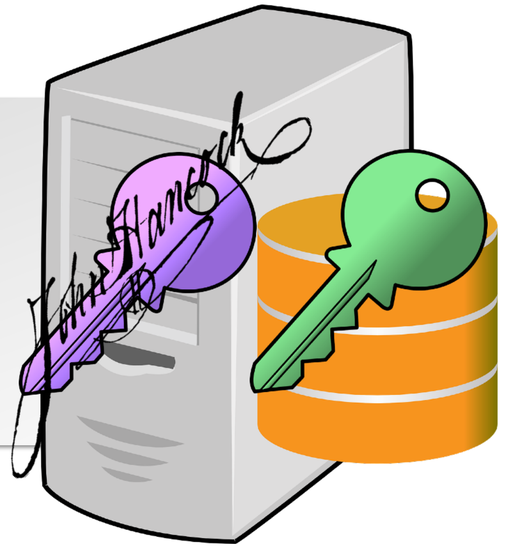
Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange





DNS lookup for server

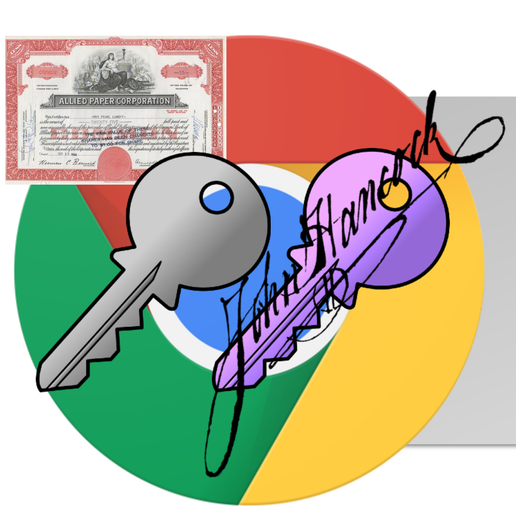
Client connects to server via TCP

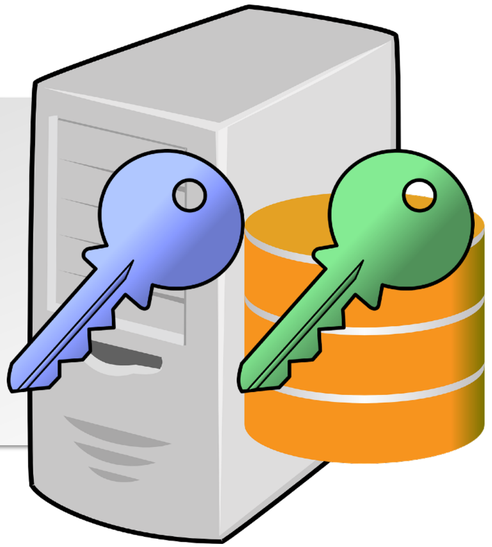
Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange





DNS lookup for server

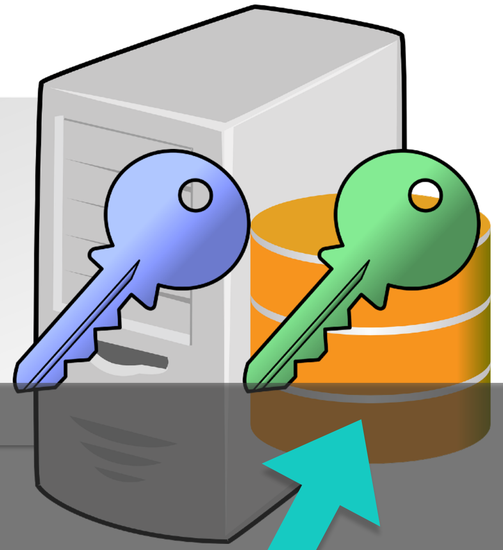
Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange



NEVER TRANSMITTED

DNS lookup for server

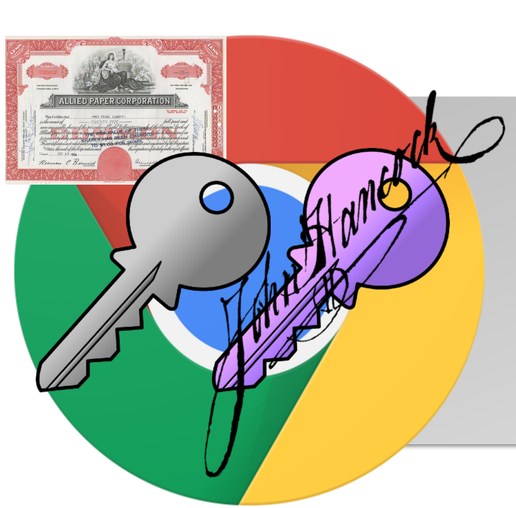
Client connects to server via TCP

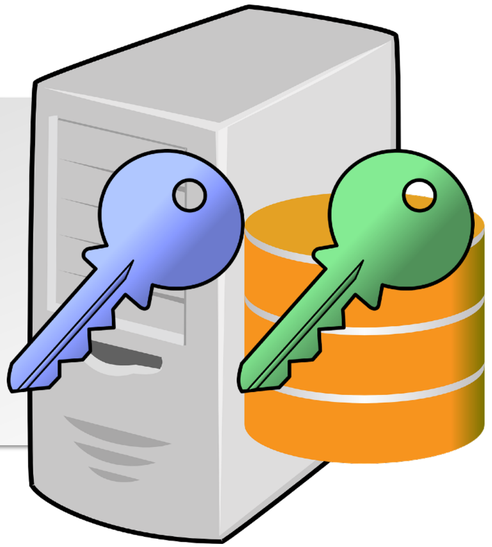
Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange





DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange

DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange

Begin encryption

DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange

Begin encryption

authenticity

privacy

integrity

DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange

Begin encryption

DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange

Begin encryption

Send login credentials to server

Server verifies login credentials

Log user in

DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange

Begin encryption

Send login credentials to server

Server verifies login credentials

Log user in

DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange

Begin encryption

Send login credentials to server

Server verifies login credentials

Log user in

transport stuff

DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange

Begin encryption

Send login credentials to server

Server verifies login credentials

Log user in

transport stuff

app stuff

DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange

Begin encryption

Send login credentials to server

Server verifies login credentials

Log user in

stuff implemented in Apache, nginx, Chrome, Safari, Firefox…

stuff in my JS, my HTML, my Gemfile, my Rails controllers

```
Send login credentials to server
Server verifies login credentials
Log user in
```

stuff in my JS, my HTML, my Gemfile, my Rails controllers

SERVER

```
] rails s

=> Booting Puma
=> Rails 5.0.1 application starting in development on http://localhost:3000
=> Run `rails server -h` for more startup options
* Version 3.7.0 (ruby 2.4.0-p0), codename: Snowy Sagebrush
* Min threads: 5, max threads: 5
* Environment: development
* Listening on tcp://0.0.0.0:3000
* Use Ctrl-C to stop
```

BROWSER

```
] telnet 127.0.0.1 3000
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
```

BROWSER

```
] telnet 127.0.0.1 3000
Trying 127.0.0.1...
Connected to localhost.
Escape character is ‘^]'.
POST /users/sign_in HTTP/1.1
Host: 127.0.0.1:3000
User-Agent: telnet
Accept: */*
Content-Length: 65
Content-Type: application/x-www-form-urlencoded

user%5Bemail%5D=railsconf%40example.com&user%5Bpassword%5D=abc123
```

DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange

Begin encryption

Send login credentials to server

Server verifies login credentials

Log user in

DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange

Begin encryption

Send login credentials to server

Server verifies login credentials

Log user in

SERVER

```
Started POST "/users/sign_in" for 127.0.0.1 at 2017-04-04 19:40:43 -0400
Processing by Devise::SessionsController#create as HTML
  Parameters: {"user"=>{"email"=>"railsconf@example.com",
"password"=>"[FILTERED]", "remember_me"=>"0"}}
  User Load (0.1ms)  SELECT  "users".* FROM "users" WHERE "users"."email" = ?
ORDER BY "users"."id" ASC LIMIT ?  [["email", "railsconf@example.com"],
["LIMIT", 1]]
   (0.1ms)  begin transaction
  SQL (0.3ms)  UPDATE "users" SET "current_sign_in_at" = ?, "last_sign_in_at"
= ?, "sign_in_count" = ?, "updated_at" = ? WHERE "users"."id" = ?
[["current_sign_in_at", 2017-04-04 23:40:43 UTC], ["last_sign_in_at",
2017-04-04 23:25:53 UTC], ["sign_in_count", 16], ["updated_at", 2017-04-04
23:40:43 UTC], ["id", 1]]
   (1.4ms)  commit transaction
Redirected to http://localhost:3000/
Completed 302 Found in 131ms (ActiveRecord: 1.8ms)
```

SERVER

```
Started POST "/users/sign_in" for 127.0.0.1 at 2017-04-04 19:40:43 -0400
Processing by Devise::SessionsController#create as HTML
  Parameters: {"user"=>{"email"=>"railsconf@example.com",
"password"=>"[FILTERED]", "remember_me"=>"0"}}
  User Load (0.1ms)  SELECT  "users".* FROM "users" WHERE "users"."email" = ?
ORDER BY "users"."id" ASC LIMIT ?  [["email", "railsconf@example.com"],
["LIMIT", 1]]
   (0.1ms)  begin transaction
  SQL (0.3ms)  UPDATE "users" SET "current_sign_in_at" = ?, "last_sign_in_at"
= ?, "sign_in_count" = ?, "updated_at" = ? WHERE "users"."id" = ?
[["current_sign_in_at", 2017-04-04 23:40:43 UTC], ["last_sign_in_at",
2017-04-04 23:25:53 UTC], ["sign_in_count", 16], ["updated_at", 2017-04-04
23:40:43 UTC], ["id", 1]]
   (1.4ms)  commit transaction
Redirected to http://localhost:3000/
Completed 302 Found in 131ms (ActiveRecord: 1.8ms)
```

SERVER

```
Started POST "/users/sign_in" for 127.0.0.1 at 2017-04-04 19:40:43 -0400
Processing by Devise::SessionsController#create as HTML
  Parameters: {"user"=>{"email"=>"railsconf@example.com",
"password"=>"[FILTERED]", "remember_me"=>"0"}}
  User Load (0.1ms)  SELECT  "users".* FROM "users" WHERE "users"."email" = ?
ORDER BY "users"."id" ASC LIMIT ?  [["email", "railsconf@example.com"],
["LIMIT", 1]]
   (0.1ms)  begin transaction
  SQL (0.3ms)  UPDATE "users" SET "current_sign_in_at" = ?, "last_sign_in_at"
= ?, "sign_in_count" = ?, "updated_at" = ? WHERE "users"."id" = ?
[["current_sign_in_at", 2017-04-04 23:40:43 UTC], ["last_sign_in_at",
2017-04-04 23:25:53 UTC], ["sign_in_count", 16], ["updated_at", 2017-04-04
23:40:43 UTC], ["id", 1]]
   (1.4ms)  commit transaction
Redirected to http://localhost:3000/
Completed 302 Found in 131ms (ActiveRecord: 1.8ms)
```


DO NOT
REMOVE CARDS
FROM TRAYS
DO NOT
REMOVE CARDS
FROM TRAYS

PASSWORDS

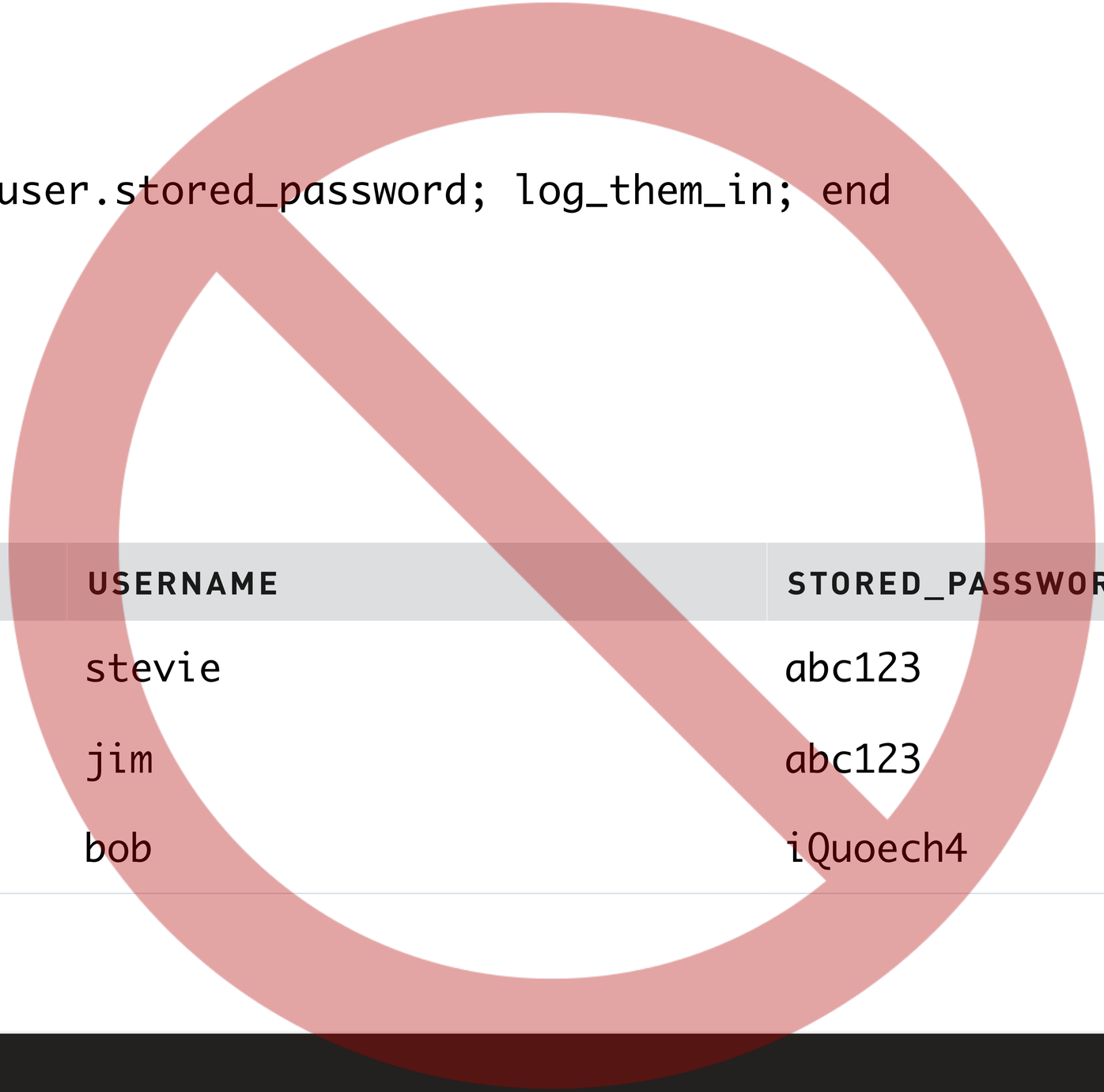
ID	USERNAME	ACTUAL PASSWORD
1	stevie	abc123
2	jim	abc123
3	bob	iQuoech4

```
if "abc123" == user.stored_password; log_them_in; end
```

```
if "abc123" == user.stored_password; log_them_in; end
```

ID	USERNAME	STORED_PASSWORD
1	stevie	abc123
2	jim	abc123
3	bob	iQuoech4

```
if "abc123" == user.stored_password; log_them_in; end
```



ID	USERNAME	STORED_PASSWORD
1	stevie	abc123
2	jim	abc123
3	bob	iQuoech4

```
if Digest::SHA1.hexdigest(“abc123”) == user.hashed_password;
    log_them_in
end
```

```
if Digest::SHA1.hexdigest(“abc123”) == user.hashed_password;
    log_them_in
end
```

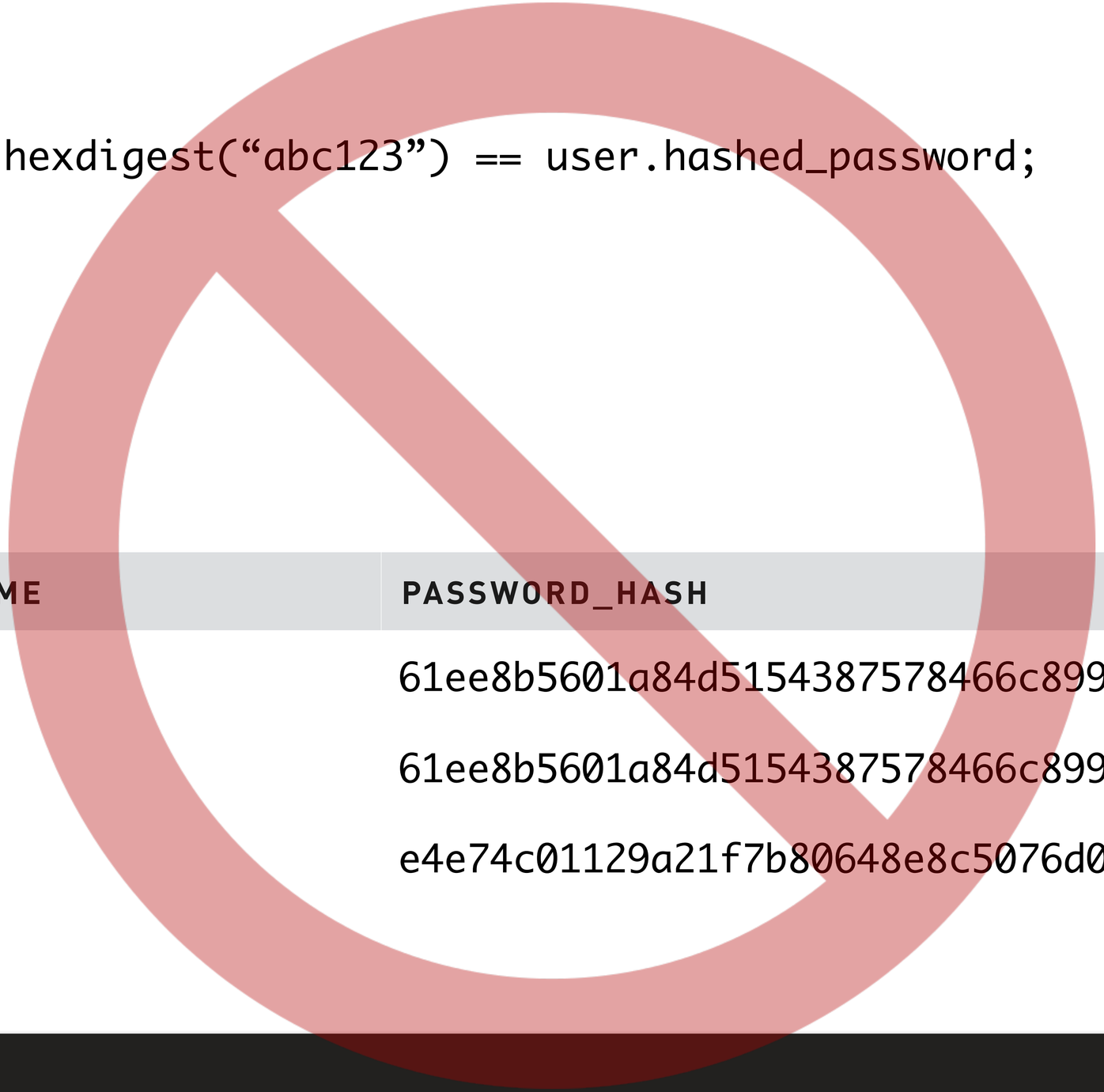
ID	USERNAME	PASSWORD_HASH
1	stevie	61ee8b5601a84d5154387578466c8998848ba089
2	jim	61ee8b5601a84d5154387578466c8998848ba089
3	bob	e4e74c01129a21f7b80648e8c5076d068f4e2e0f

```
if Digest::SHA1.hexdigest(“abc123”) == user.hashed_password;
    log_them_in
end
```

ID	USERNAME	PASSWORD_HASH
1	stevie	61ee8b5601a84d5154387578466c8998848ba089
2	jim	61ee8b5601a84d5154387578466c8998848ba089
3	bob	e4e74c01129a21f7b80648e8c5076d068f4e2e0f

```
if Digest::SHA1.hexdigest(“abc123”) == user.hashed_password;
    log_them_in
end
```

ID	USERNAME	PASSWORD_HASH
1	stevie	61ee8b5601a84d5154387578466c8998848ba089
2	jim	61ee8b5601a84d5154387578466c8998848ba089
3	bob	e4e74c01129a21f7b80648e8c5076d068f4e2e0f



```
if Digest::SHA1.hexdigest(“#{user.salt}-abc123”) == user.hashed_password;
    log_them_in
end
```

```
if Digest::SHA1.hexdigest(“#{user.salt}-abc123”) == user.hashed_password;
    log_them_in
end
```

ID	USERNAME	PASSWORD_HASH	SALT
1	stevie	e028326ea98cbd99dfcaa7a901b74ae518f61919	riet9ooM
2	jim	d23b41001c014fa3a7a402158c67df7b4c6ca274	nohkon9T
3	bob	396101deb9c98c846367ae6988cab229f9d459f6	QuiT8wei

```
if Digest::SHA1.hexdigest(“#{user.salt}-abc123”) == user.hashed_password;
    log_them_in
end
```

ID	USERNAME	PASSWORD_HASH	SALT
1	stevie	e028326ea98cbd99dfcaa7a901b74ae518f61919	riet9ooM
2	jim	d23b41001c014fa3a7a402158c67df7b4c6ca274	nohkon9T
3	bob	396101deb9c98c846367ae6988cab229f9d459f6	QuiT8wei

```
if BCrypt::Password.create(user.password_hash) == “abc123”
    log_them_in
end
```

```
if BCrypt::Password.create(user.password_hash) == “abc123”
    log_them_in
end
```

ID	USERNAME	PASSWORD_HASH
1	stevie	$2a$12$OhHcuM6JnoA7144ea6FmEuD737.kisLq.5mZATrg2bSkF1jRnjfV.
2	jim	$2a$12$YcZpwbDB8R26C7HHqPCYne3ATojc3kLEhkomV6z4GXfYgJZuItnAa
3	bob	$2a$12$jABxCCxO32TW.fJvo.RZOeaQW7hBEFhOw0Y8.U0t0soaDM7/Z6W4q

```
if BCrypt::Password.create(user.password_hash) == “abc123”
    log_them_in
end
```

```
$2a$12$OhHcuM6JnoA7144ea6FmEuD737.kisLq.5mZATrg2bSkF1jRnjfV.
```

```
if BCrypt::Password.create(user.password_hash) == “abc123”
    log_them_in
end
```

```
$2a$12$OhHcuM6JnoA7144ea6FmEuD737.kisLq.5mZATrg2bSkF1jRnjfV.
```

```
if BCrypt::Password.create(user.password_hash) == “abc123”
    log_them_in
end
```

```
$2a$12$OhHcuM6JnoA7144ea6FmEuD737.kisLq.5mZATrg2bSkF1jRnjfV.
```

```
if BCrypt::Password.create(user.password_hash) == “abc123”
    log_them_in
end
```

$2a$12$0hHcuM6JnoA7144ea6FmEuD737.kisLq.5mZATrg2bSkF1jRnjfV.

```
if BCrypt::Password.create(user.password_hash) == “abc123”
    log_them_in
end
```

$2a$12$OhHcuM6JnoA7144ea6FmEuD737.kisLq.5mZATrg2bSkF1jRnjfV.

DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange

Begin encryption

Send login credentials to server

Server verifies login credentials

Log user in

BROWSER

```
] telnet 127.0.0.1 3000
Trying 127.0.0.1...
Connected to localhost.
Escape character is ‘^]'.
POST /users/sign_in HTTP/1.1
Host: 127.0.0.1:3000
User-Agent: telnet
Accept: */*
Content-Length: 65
Content-Type: application/x-www-form-urlencoded

user%5Bemail%5D=railsconf%40example.com&user%5Bpassword%5D=abc123
```

```
] telnet 127.0.0.1 3000
Trying 127.0.0.1...
Connected to localhost.
Escape character is ‘^]'.
POST /users/sign_in HTTP/1.1
Host: 127.0.0.1:3000
User-Agent: telnet
Accept: */*
Content-Length: 65
Content-Type: application/x-www-form-urlencoded

user%5Bemail%5D=railsconf%40example.com&user%5Bpassword%5D=abc123

HTTP/1.1 302 Found
X-Frame-Options: SAMEORIGIN
X-XSS-Protection: 1; mode=block
```

BROWSER

```
HTTP/1.1 302 Found
X-Frame-Options: SAMEORIGIN
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
Location: http://127.0.0.1:3000/
Content-Type: text/html; charset=utf-8
Cache-Control: no-cache
Set-Cookie: _login-example_session=d28ded87eca5fd708bc53e66623db73b; path=/;
HttpOnly
X-Request-Id: b17461fb-09ea-4eb4-bf6f-9e625aca427c
X-Runtime: 0.150167
Transfer-Encoding: chunked

58
<html><body>You are being <a href="http://127.0.0.1:3000/">redirected</a>.</
body></html>
```

BROWSER

```
HTTP/1.1 302 Found
X-Frame-Options: SAMEORIGIN
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
Location: http://127.0.0.1:3000/
Content-Type: text/html; charset=utf-8
Cache-Control: no-cac
Set-Cookie: _login-example_session=d28ded87eca5fd708bc53e66623db73b; path=/;
HttpOnly
X-Request-Id: b17461fb-09ea-4eb4-bf6f-9e625aca427c
X-Runtime: 0.150167
Transfer-Encoding: chunked

58
<html><body>You are being <a href="http://127.0.0.1:3000/">redirected</a>.</
body></html>
```

DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange

Begin encryption

Send login credentials to server

Server verifies login credentials

Log user in

PROPERTIES OF SECURE SESSIONS

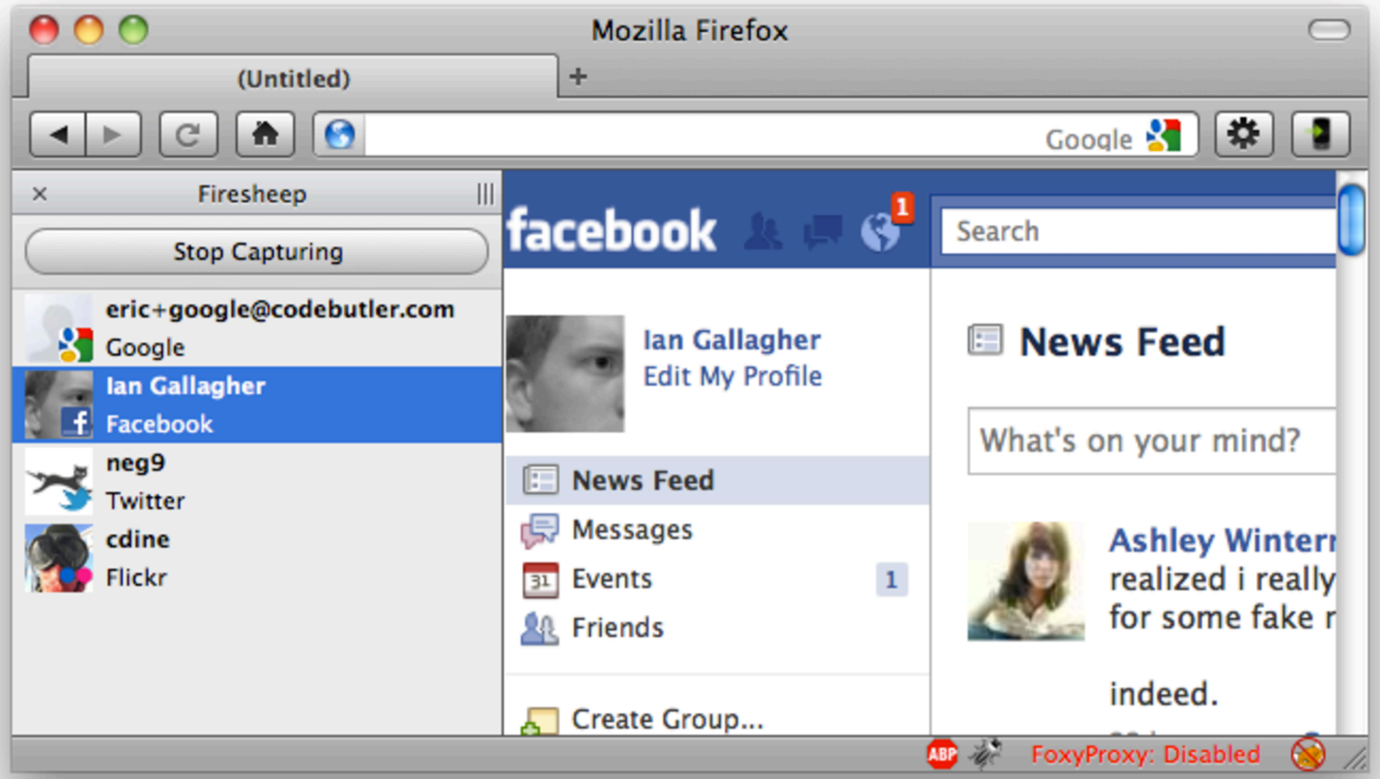
- Authenticate the person (i.e. by username and password)
- And we issued an unpredictable, unique session token.
- And we know nobody else has the token because it was sent on a secure channel.


[redacted] at
20 hrs
Winner winner chicken dinner!!


[redacted] at
20 hrs
Winner winner chicken dinner!!


[redacted] at
20 hrs
Winner winner chicken dinner!!

Double-click on someone, and you're instantly logged in as them.



That's it.

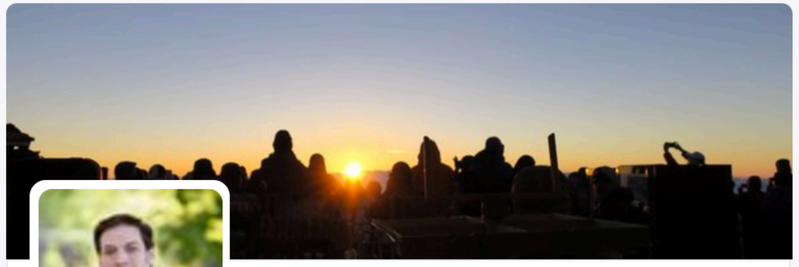
(http://codebutler.com/firesheep)

PROPERTIES OF SECURE SESSIONS

- Authenticate the person (i.e. by username and password)
- And we issued an unpredictable, unique session token.
- And we know nobody else has the token because it was sent on a secure channel.

PROPERTIES OF SECURE SESSIONS

- Authenticate the person (i.e. by username and password)
- And we issued an unpredictable, unique session token.
- ~~And we know nobody else has the token because it was sent on a secure channel.~~



swieton
@swieton

TWEETS	FOLLOWING	FOLLOWERS
14	36	31

Trends · Change

Dennis Rodman

#LHHATL
41.3K Tweets

Palace
168K Tweets

#SuperstarShakeUp
52.3K Tweets

Pulitzer
The 2017 Pulitzer Prize winners have been announced

What's happening?

FuzyPinkBunny @FuzyPinkBunny · 7m
My first attempt at #pixelart This is a snelpling. A small gelatinous creature (hence the no arms)

1 1

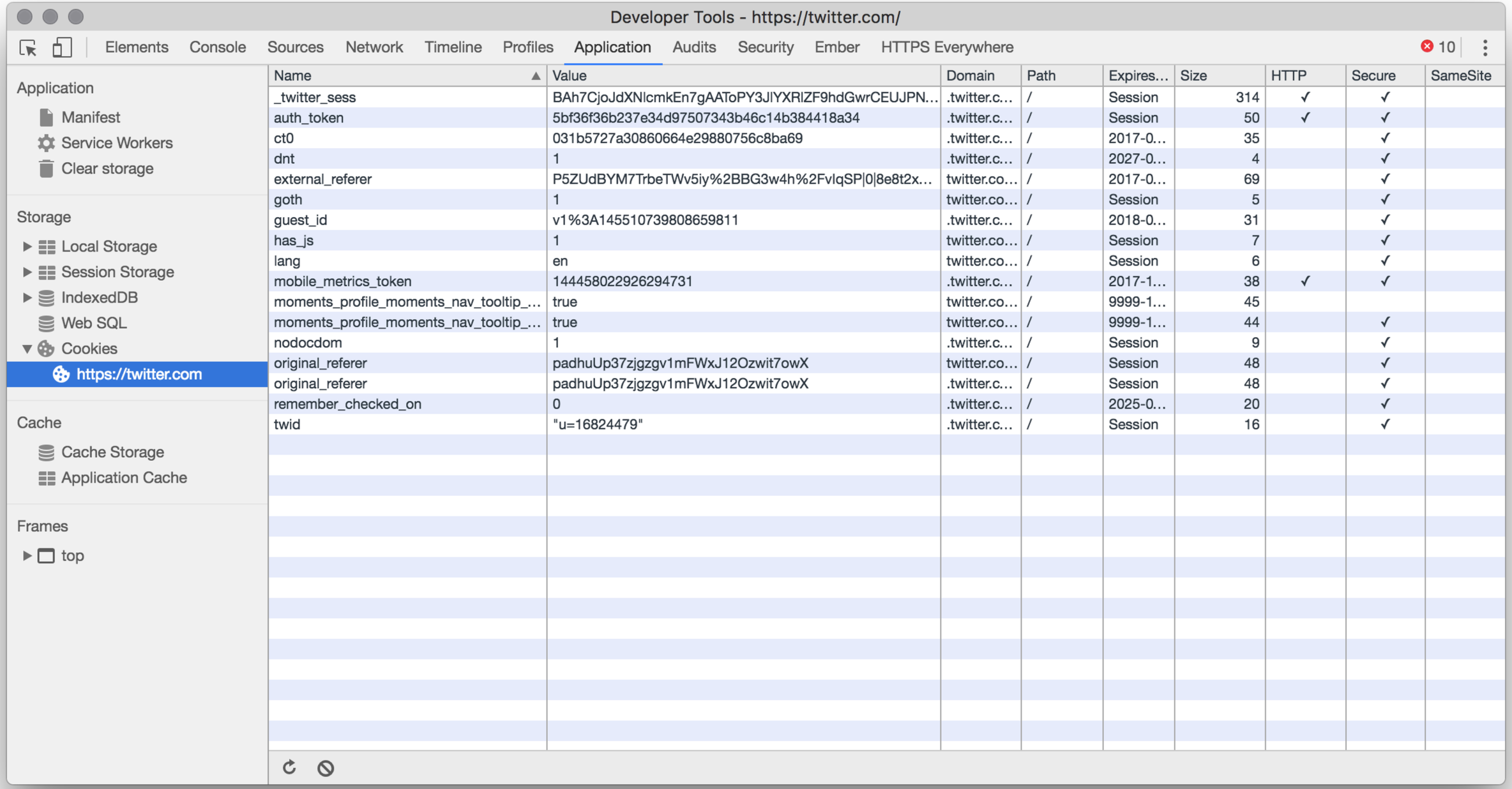
In case you missed it

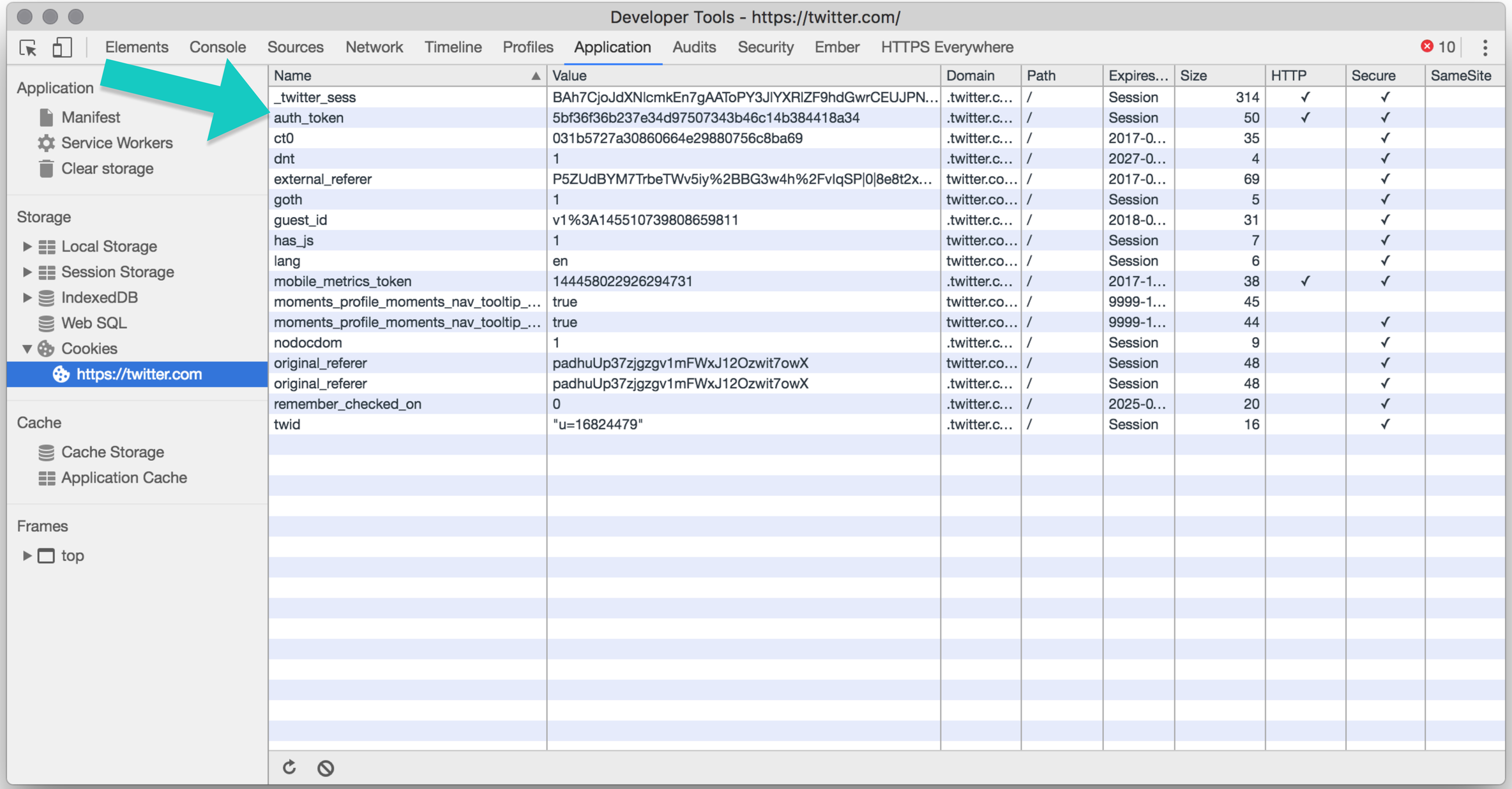
RailsConf @railsconf · 3h
We have now officially sold out all of our RailsConf 2017 reserved hotel blocks. Thank you, and see you all in a few weeks in Phoenix!

12

Krista Nelson Retweeted

The Practical Dev @ThePracticalDev · 12h
Opening your code editor on Monday morning


Developer Tools - https://twitter.com/
Elements
Console
Sources
Network
Timeline
Profiles
Application
Audits
Security
Ember
HTTPS Everywhere
10
Application
Manifest
Service Workers
Clear storage
Storage
Local Storage
Session Storage
IndexedDB
Web SQL
Cookies
https://twitter.com
Cache
Cache Storage
Application Cache
Frames
top
Name
Value
Domain
Path
Expires...
Size
HTTP
Secure
SameSite
_twitter_sess
BAh7CjoJdXNlcmkEn7gAAToPY3JlYXRlZF9hdGwrCEUJPN...
.twitter.c...
/
Session
314
✓
✓
auth_token
5bf36f36b237e34d97507343b46c14b384418a34
.twitter.c...
/
Session
50
✓
✓
ct0
031b5727a30860664e29880756c8ba69
.twitter.c...
/
2017-0...
35
✓
dnt
1
.twitter.c...
/
2027-0...
4
✓
external_referer
P5ZUdBYM7TrbeTWv5iy%2BBG3w4h%2FvlqSP|0|8e8t2x...
twitter.co...
/
2017-0...
69
✓
goth
1
twitter.co...
/
Session
5
✓
guest_id
v1%3A145510739808659811
.twitter.c...
/
2018-0...
31
✓
has_js
1
twitter.co...
/
Session
7
✓
lang
en
twitter.co...
/
Session
6
✓
mobile_metrics_token
144458022926294731
.twitter.c...
/
2017-1...
38
✓
✓
moments_profile_moments_nav_tooltip_...
true
twitter.co...
/
9999-1...
45
moments_profile_moments_nav_tooltip_...
true
twitter.co...
/
9999-1...
44
✓
nodocdom
1
.twitter.c...
/
Session
9
✓
original_referer
padhuUp37zjgzgv1mFWxJ12Ozwit7owX
twitter.co...
/
Session
48
✓
original_referer
padhuUp37zjgzgv1mFWxJ12Ozwit7owX
.twitter.c...
/
Session
48
✓
remember_checked_on
0
.twitter.c...
/
2025-0...
20
✓
twid
"u=16824479"
.twitter.c...
/
Session
16
✓


Developer Tools - https://twitter.com/
Elements
Console
Sources
Network
Timeline
Profiles
Application
Audits
Security
Ember
HTTPS Everywhere
10
Application
Manifest
Service Workers
Clear storage
Storage
Local Storage
Session Storage
IndexedDB
Web SQL
Cookies
https://twitter.com
Cache
Cache Storage
Application Cache
Frames
top
Name
Value
Domain
Path
Expires...
Size
HTTP
Secure
SameSite
_twitter_sess
BAh7CjoJdXNlcmkEn7gAAToPY3JlYXRlZF9hdGwrCEUJPN...
.twitter.c...
/
Session
314
✓
✓
auth_token
5bf36f36b237e34d97507343b46c14b384418a34
.twitter.c...
/
Session
50
✓
✓
ct0
031b5727a30860664e29880756c8ba69
.twitter.c...
/
2017-0...
35
✓
dnt
1
.twitter.c...
/
2027-0...
4
✓
external_referer
P5ZUdBYM7TrbeTWv5iy%2BBG3w4h%2FvlqSP|0|8e8t2x...
twitter.co...
/
2017-0...
69
✓
goth
1
twitter.co...
/
Session
5
✓
guest_id
v1%3A145510739808659811
.twitter.c...
/
2018-0...
31
✓
has_js
1
twitter.co...
/
Session
7
✓
lang
en
twitter.co...
/
Session
6
✓
mobile_metrics_token
144458022926294731
.twitter.c...
/
2017-1...
38
✓
✓
moments_profile_moments_nav_tooltip_...
true
twitter.co...
/
9999-1...
45
moments_profile_moments_nav_tooltip_...
true
twitter.co...
/
9999-1...
44
✓
nodocdom
1
.twitter.c...
/
Session
9
✓
original_referer
padhuUp37zjgzgv1mFWxJ12Ozwit7owX
twitter.co...
/
Session
48
✓
original_referer
padhuUp37zjgzgv1mFWxJ12Ozwit7owX
.twitter.c...
/
Session
48
✓
remember_checked_on
0
.twitter.c...
/
2025-0...
20
✓
twid
"u=16824479"
.twitter.c...
/
Session
16
✓

```
] curl -v -H "Cookie: auth_token=5bf36f36b237e34d97507343b46c14b384418a34”\
https://twitter.com/
```

```
] curl -v -H "Cookie: auth_token=5bf36f36b237e34d97507343b46c14b384418a34”\
https://twitter.com/
*   Trying 104.244.42.65...
* TCP_NODELAY set
* Connected to twitter.com (104.244.42.65) port 443 (#0)
* TLS 1.2 connection using TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
* Server certificate: twitter.com
* Server certificate: DigiCert SHA2 Extended Validation Server CA
* Server certificate: DigiCert High Assurance EV Root CA
> GET / HTTP/1.1
> Host: twitter.com
> User-Agent: curl/7.51.0
> Accept: */*
> Cookie: auth_token=5bf36f36b237e34d97507343b46c14b384418a34
>
< HTTP/1.1 200 OK
```

```
< HTTP/1.1 200 OK
< cache-control: no-cache, no-store, must-revalidate, pre-check=0, post-check=0
< content-type: text/html;charset=utf-8
< date: Sun, 09 Apr 2017 14:33:38 GMT
< expires: Tue, 31 Mar 1981 05:00:00 GMT
< last-modified: Sun, 09 Apr 2017 14:33:38 GMT
< pragma: no-cache
< server: tsa_b
< set-cookie: dnt=1; Expires=Wed, 07 Apr 2027 14:33:38 UTC; Path=/; Domain=.twitter.com
< set-cookie: fm=0; Expires=Sun, 09 Apr 2017 14:33:28 UTC; Path=/; Domain=.twitter.com; Secure; HTTPOnly
< set-cookie: _twitter_sess=BAh7CSIKZmxhc2hJQzonQWN0aW9uQ29udHJvbGxlcjo6Rmxhc2g6OkZsYXNo%250ASGFzaHsABjoKQHVzZWR7ADoPY3JlYXRlZF9hdGwrCO9xIlNbATOMY3NyZl9p%250AZCIlYTEwNzI5ZWJjY2VmNDJlNTI2MGU4MzVjZGM5ODYyNmM6B2lkIiU0ZWQw%250AZjNlYjE2Y2VlZDFhZDNkOTZkNzZiNjEwOGQ4Ng%253D%253D--8a581172ef4b4f2ac4acf09b3238cc32a50d7031; Path=/;
```

```
<!DOCTYPE html>
<html lang="en" data-scribe-reduced-action-queue="true">
  <head>
    <meta charset="utf-8">
    <noscript><meta http-equiv="refresh" content="0; URL=https://
mobile.twitter.com/i/nojs_router?path=%2F"></noscript>




  <script id="bouncer_terminate_iframe" nonce="dTCdbjC6TYfNBq5K47uWMw==">
    if (window.top != window) {
  window.top.postMessage({'bouncer': true, 'event': 'complete'}, '*');
}
  </script>
  <script id="resolve_inline_redirects" nonce="dTCdbjC6TYfNBq5K47uWMw==">
    !function(){function n(){var n=window.location.href.match(/#(.)(.*)
$/);return n&&"!"==n[1]&&n[2].replace(/^\//,"")}function g(n){return!!
```

```
<link rel="search" type="application/opensearchdescription+xml" href="/
opensearch.xml" title="Twitter">


    <link id="async-css-placeholder">


<style id="user-style-swieton">
```

```
<link rel="search" type="application/opensearchdescription+xml" href="/
opensearch.xml" title="Twitter">


    <link id="async-css-placeholder">


<style id="user-style-swieton">
```

```
] curl -v -H "Cookie: auth_token=5bf36f36b237e34d97507343b46c14b384418a34”\
https://twitter.com/
```

DNS lookup for server

Client connects to server via TCP

Server sends certificate

Client verifies CA signature

Client verifies certificate matches domain

Session key exchange

Begin encryption

Send login credentials to server

Server verifies login credentials

Log user in

“Wait, who are you?”

EMPLOYEE DATABASE

WIKI

EMPLOYEE DATABASE

WIKI.COM

EMPLOYEES.COM

WIKI.COM

EMPLOYEES.COM

WIKI.COM

EMPLOYEES.COM

WIKI.COM

EMPLOYEES.COM

WIKI.COM

EMPLOYEES.COM

WIKI.COM

EMPLOYEES.COM

“Gimme the datas!”

`wiki.com/index`

WIKI.COM

EMPLOYEES.COM

“Gimme the datas!”

`wiki.com/index`

“But I don’t know you!”

WIKI.COM

EMPLOYEES.COM

“Gimme the datas!”

`wiki.com/index`

“But I don’t know you!”

“… Go talk to that guy.”

WIKI.COM

EMPLOYEES.COM

“Gimme the datas!”

`wiki.com/index`

“But I don’t know you!”

“… Go talk to that guy.”

`302 Redirect: employees.com/authenticate?returnTo=wiki.com%2Findex`

WIKI.COM

EMPLOYEES.COM

WIKI.COM

EMPLOYEES.COM

WIKI.COM

EMPLOYEES.COM

“Tell him to gimme the datas!”

`employees.com/authenticate?returnTo=wiki.com%2Findex`

WIKI.COM

EMPLOYEES.COM

“Tell him to gimme the datas!”

`employees.com/authenticate?returnTo=wiki.com%2Findex`

“But I don’t know you!”

WIKI.COM

EMPLOYEES.COM

“Tell him to gimme the datas!”

`employees.com/authenticate?returnTo=wiki.com%2Findex`

“But I don’t know you!”

“… Do you have credentials?”

WIKI.COM

EMPLOYEES.COM

“Tell him to gimme the datas!”

`employees.com/authenticate?returnTo=wiki.com%2Findex`

“But I don’t know you!”

“… Do you have credentials?”

`<log in form!>`

WIKI.COM

EMPLOYEES.COM

WIKI.COM

EMPLOYEES.COM

“My name is Michael!”

```
POST employees.com/login?returnTo=wiki.com%2Findex&username=…
```

WIKI.COM

EMPLOYEES.COM

“My name is Michael!”

```
POST employees.com/login?returnTo=wiki.com%2Findex&username=…
```

“Ok, I believe you.”

WIKI.COM

EMPLOYEES.COM

“My name is Michael!”

```
POST employees.com/login?returnTo=wiki.com%2Findex&username=…
```

“Ok, I believe you.”

“Here’s a cookie for next time!”

WIKI.COM

EMPLOYEES.COM

“My name is Michael!”

```
POST employees.com/login?returnTo=wiki.com%2Findex&username=…
```

“Ok, I believe you.”

“Here’s a cookie for next time!”

“And take this token over to the wiki for now.”

WIKI.COM

EMPLOYEES.COM

THE TOKEN

Two parts:

- Who I am
- An authentication code to prove:
  - it came from someone who can vouch for me
  - and it hasn’t been modified

THE TOKEN - JSON

```
{user_id: 15,
username: “stevie",
role: “admin",
expires_at: <30 seconds from now>}
```

THE TOKEN - URL-ENCODED JSON

%7Buser_id%3A%2015%2C%20username%3A%20%22stevie%22%2C%20role%3A%20%22admin%22%2C%20expires_in%3A%20%2230%20seconds%22%7D

THE TOKEN

Two parts:

- Who I am ✓
- An authentication code to prove:
  - it came from someone who can vouch for me
  - and it hasn’t been modified

THE HMAC

```
## Hash-based message authentication code:

key = "cats!"
data = "%7Buser_id%3A%2015%2C%20username%3A%20%22stevie%22%2C%20role%3A%
hmac = OpenSSL::HMAC.hexdigest(OpenSSL::Digest.new('sha1'), key, data)

# hmac == “49466927946586 8f35fed9fe63f4d57ca768bdeb”
```

“My name is Michael!”

```
POST employees.com/login?returnTo=wiki.com%2FisOk&username=…
```

WIKI.COM

EMPLOYEES.COM

“My name is Michael!”

```
POST employees.com/login?returnTo=wiki.com%2FisOk&username=…
```

“Ok, I believe you.”

“Here’s a cookie for next time!”

“And take this token over to the wiki for now.”

WIKI.COM

EMPLOYEES.COM

“My name is Michael!”

```
POST employees.com/login?returnTo=wiki.com%2FisOk&username=…
```

“Ok, I believe you.”

“Here’s a cookie for next time!”

“And take this token over to the wiki for now.”

```
302 Redirect: wiki.com/isOk?auth=%7Buser_id%3A%2015%2C%20username%3A%20%22stevie…
        &hmac=49466927946586 8f35fed9fe63f4d57ca768bdeb&returnTo=wiki.com%2Findex"
```

WIKI.COM

EMPLOYEES.COM

“My name is Michael!”

```
POST employees.com/login?returnTo=wiki.com%2FisOk&username=…
```

“Ok, I believe you.”

“Here’s a cookie for next time!”

“And take this token over to the wiki for now.”

```
302 Redirect: wiki.com/isOk?auth=%7Buser_id%3A%2015%2C%20username%3A%20%22stevie…
        &hmac=4946692794658868f35fed9fe63f4d57ca768bdeb&returnTo=wiki.com%2Findex"
```

WIKI.COM

EMPLOYEES.COM

“My name is Michael!”

```
POST employees.com/login?returnTo=wiki.com%2FisOk&username=…
```

“Ok, I believe you.”

“Here’s a cookie for next time!”

“And take this token over to the wiki for now.”

```
302 Redirect: wiki.com/isOk?auth=%7Buser_id%3A%2015%2C%20username%3A%20%22stevie…
        &hmac=49466927946586 8f35fed9fe63f4d57ca768bdeb&returnTo=wiki.com%2Findex"
```

WIKI.COM

EMPLOYEES.COM

“My name is Michael!”

```
POST employees.com/login?returnTo=wiki.com%2FisOk&username=…
```

“Ok, I believe you.”

“Here’s a cookie for next time!”

“And take this token over to the wiki for now.”

```
302 Redirect: wiki.com/isOk?auth=%7Buser_id%3A%2015%2C%20username%3A%20%22stevie…
        &hmac=4946692794658681351ed9fe631f4a57ca768baeb&returnTo=wiki.com%2Findex
```

WIKI.COM

EMPLOYEES.COM

“My name is Michael!”

```
POST employees.com/login?returnTo=wiki.com%2FisOk&username=…
```

“Ok, I believe you.”

“Here’s a cookie for next time!”

“And take this token over to the wiki for now.”

```
302 Redirect: wiki.com/isOk?auth=%7Buser_id%3A%2015%2C%20username%3A%20%22stevie…
        &hmac=49466927946586 8f35fed9fe63f4d57ca768bdeb&returnTo=wiki.com%2Findex"
```

WIKI.COM

EMPLOYEES.COM

“My name is Michael!”

```
POST employees.com/login?returnTo=wiki.com%2FisOk&username=…
```

“Ok, I believe you.”

“Here’s a cookie for next time!”

“And take this token over to the wiki for now.”

```
302 Redirect: wiki.com/isOk?auth=%7Buser_id%3A%2015%2C%20username%3A%20%22stevie…
        &hmac=49466927946586 8f35fed9fe63f4d57ca768bdeb&returnTo=wiki.com%2Findex"
```

WIKI.COM

EMPLOYEES.COM

WIKI.COM

EMPLOYEES.COM

“He told me this token would prove my identity.”

wiki.com/isOk?auth=%7Buser_id%3A%2015%2C%20username%3A%20%22stevie…
&hmac=49466927946586 8f35fed9fe63f4d57ca768bdeb&returnTo=wiki.com%2F
index”

WIKI.COM

EMPLOYEES.COM

“He told me this token would prove my identity.”

```
wiki.com/isOk?auth=%7Buser_id%3A%2015%2C%20username%3A%20%22stevie…
&hmac=49466927946586 8f35fed9fe63f4d57ca768bdeb&returnTo=wiki.com%2F
index”
```

“That checks out.”

“Here’s a cookie for next time.”

WIKI.COM

EMPLOYEES.COM

“He told me this token would prove my identity.”

```
wiki.com/isOk?auth=%7Buser_id%3A%2015%2C%20username%3A%20%22stevie…
&hmac=49466927946586 8f35fed9fe63f4d57ca768bdeb&returnTo=wiki.com%2F
index”
```

“That checks out.”

“Here’s a cookie for next time.”

“As you were!”

```
302 Redirect: wiki.com/index
```

WIKI.COM

EMPLOYEES.COM

WIKI.COM

EMPLOYEES.COM

“Take cookie! Now I can has dataz?”

`wiki.com/index`

WIKI.COM

EMPLOYEES.COM

“Take cookie! Now I can has dataz?”

`wiki.com/index`

“Your cookie is good.”

WIKI.COM

EMPLOYEES.COM

“Take cookie! Now I can has dataz?”

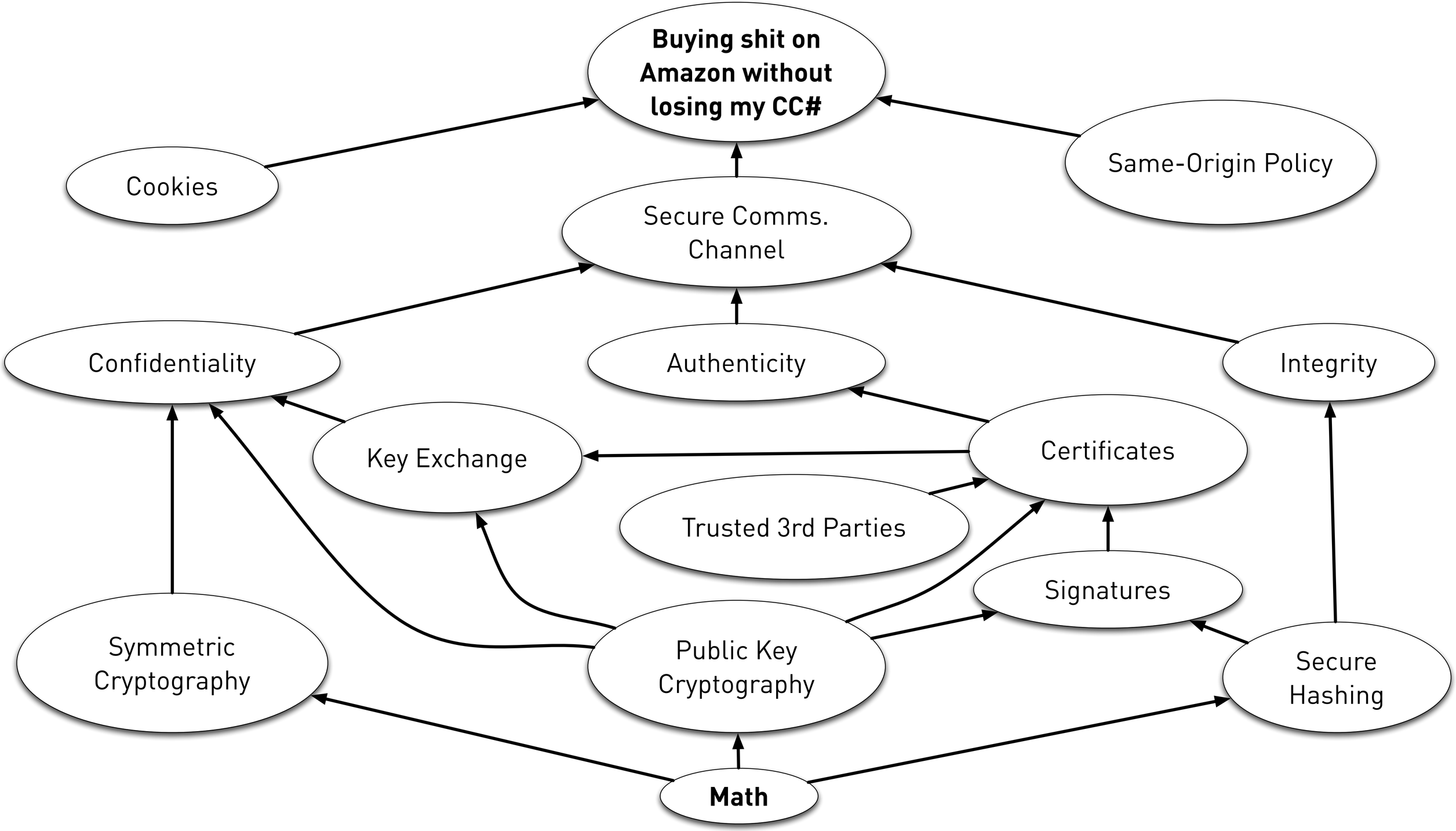
`wiki.com/index`

“Your cookie is good.”

“Here, have lots of datas!”

WIKI.COM

EMPLOYEES.COM


Buying shit on Amazon without losing my CC#
Cookies
Same-Origin Policy
Secure Comms. Channel
Confidentiality
Authenticity
Integrity
Key Exchange
Certificates
Trusted 3rd Parties
Signatures
Symmetric Cryptography
Public Key Cryptography
Secure Hashing
Math

MICHAEL SWIETON

SWIETON@ATOMICOBJECT.COM

ATOMIC OBJECT