Lecture 28:

ECE 7420 / ENGI 9807: Security

Web authentication

Web authentication

Authentication factors

TLS client certificates

OAuth

Cookies

Authentication signals

Authentication factors

Recall:

Authentication factors

Recall:

- something you *know*

Authentication factors

Recall:

- something you *know*
- something you *have*

Authentication factors

Recall:

- something you *know*
- something you *have*
  - or just something else you know?
  - something your computer knows?

Authentication factors

Recall:

- something you *know*
- something you *have*
  - or just something else you know?
  - something your computer knows?
- something you *are*

Authentication factors

Recall:

- something you *know*
- something you *have*
  - or just something else you know?
  - something your computer knows?
- something you *are* (which can be copied!)

Passwords

Passwords

Benefits

Passwords, for all of their limitations, aren't 100% bad. They are used so extensively for some sensible reasons (as well as a few poor reasons, like intertia or a lack of knowledge of alternatives on the part of software developers).

Passwords

Benefits

Problems

Passwords, for all of their limitations, aren't 100% bad. They are used so extensively for some sensible reasons (as well as a few poor reasons, like intertia or a lack of knowledge of alternatives on the part of software developers).

Passwords

Benefits

Problems

Strategies

Passwords, for all of their limitations, aren't 100% bad. They are used so extensively for some sensible reasons (as well as a few poor reasons, like intertia or a lack of knowledge of alternatives on the part of software developers).

The risks of password usage can be mitigated through sound password management strategies. What are some things that every password-verifying system ought to do?

TLS client certificates

Remember TLS?

TLS client certificates

Remember TLS?

What do CAs do?

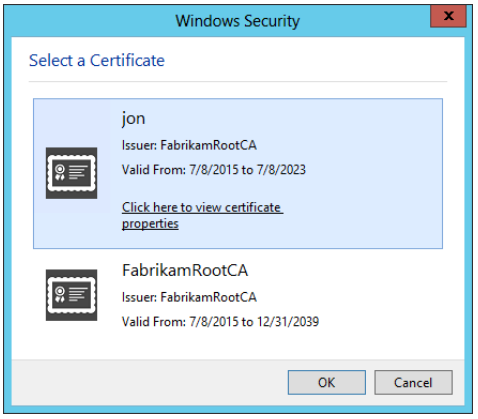
CAs mostly vouch for _____, confirming that a particular public key _____. If that public key is used to _____, we can have some assurance that _____. There are also some problems with CAs (as we've seen), but the risks associated with rogue CAs are lower than they used to be.

TLS client certificates

Remember TLS?

What do CAs do?

Can also give *clients* certificates



Source: auth0.com

CAs mostly vouch for _____, confirming that a particular public key _____. If that public key is used to _____, we can have some assurance that _____. There are also some problems with CAs (as we've seen), but the risks associated with rogue CAs are lower than they used to be.

CAs don't *only* issue server certificates, however. We've already seen _____ _____, but they can also issue client certificates! This allows both the server *and* the client to identify themselves when setting up a TLS connection.

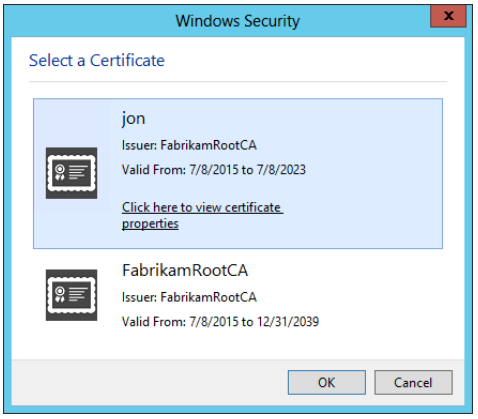
TLS client certificates

Remember TLS?

What do CAs do?

Can also give *clients* certificates

- *mutual* authentication



Source: auth0.com

CAs mostly vouch for _____, confirming that a particular public key _____. If that public key is used to _____, we can have some assurance that _____. There are also some problems with CAs (as we've seen), but the risks associated with rogue CAs are lower than they used to be.

CAs don't *only* issue server certificates, however. We've already seen _____ _____, but they can also issue client certificates! This allows both the server *and* the client to identify themselves when setting up a TLS connection.

This form of mutual authentication, based on _____, is much stronger than _____. It also has the benefit (and the cost!) of preventing users from _____.

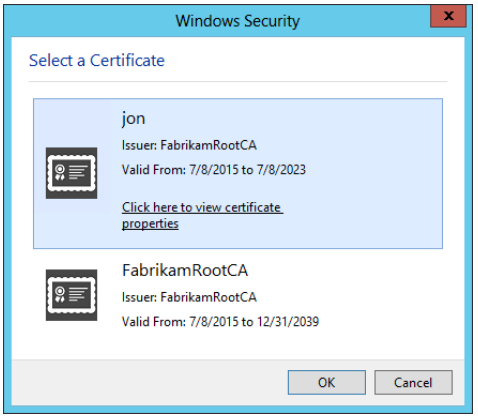
TLS client certificates

Remember TLS?

What do CAs do?

Can also give *clients* certificates

- *mutual* authentication
- a few challenges...



Source: auth0.com

CAs mostly vouch for _____, confirming that a particular public key _____. If that public key is used to _____, we can have some assurance that _____. There are also some problems with CAs (as we've seen), but the risks associated with rogue CAs are lower than they used to be.

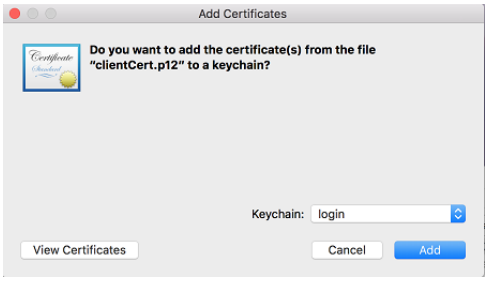
CAs don't *only* issue server certificates, however. We've already seen _____ _____, but they can also issue client certificates! This allows both the server *and* the client to identify themselves when setting up a TLS connection.

This form of mutual authentication, based on _____, is much stronger than _____. It also has the benefit (and the cost!) of preventing users from _____.

Client certificate issues

Client certificate issues

User experience

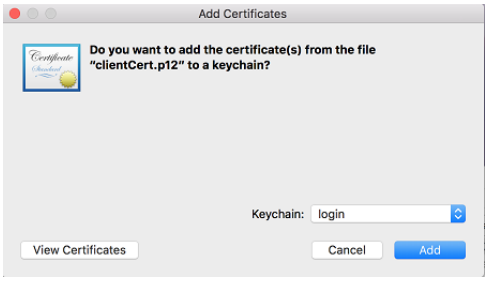


Source: Microsoft

Client certificate issues

User experience

- unusual



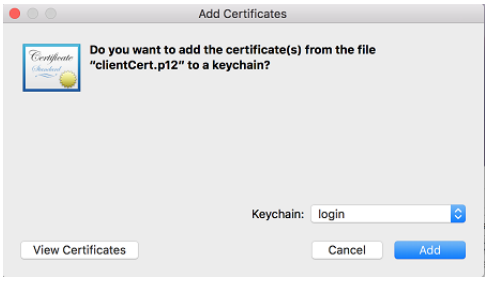
Source: Microsoft

It's very unusual to be prompted for a TLS client certificate. In fact, it's even unusual to see a browser-native password prompt rather than an HTML form:

Client certificate issues

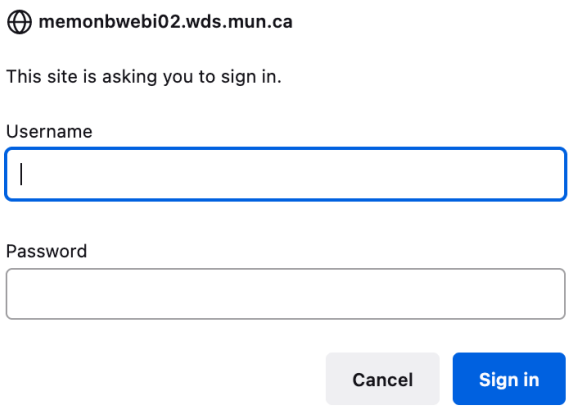
User experience

- unusual
- pre-interaction



Source: Microsoft

It's very unusual to be prompted for a TLS client certificate. In fact, it's even unusual to see a browser-native password prompt rather than an HTML form:



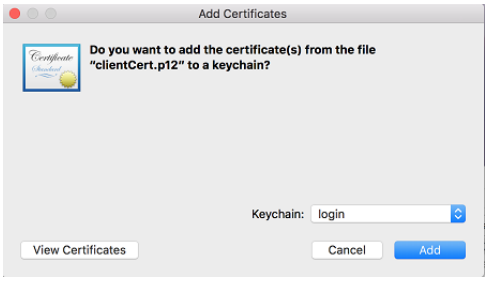
One reason for this is that it doesn't give the user a chance to see the website that they're logging into. When you visit ecorp.com, they don't want you to see a generic browser prompt, they want you to see the eCorp logo and feel warm fuzzies about their brand. This is true of HTTP `Basic-Auth` authentication, and it's also true of TLS client certificate authentication.

Client certificate issues

User experience

- unusual
- pre-interaction

TLS termination and trust



Source: Microsoft

It's very unusual to be prompted for a TLS client certificate. In fact, it's even unusual to see a browser-native password prompt rather than an HTML form:

One reason for this is that it doesn't give the user a chance to see the website that they're logging into. When you visit ecorp.com, they don't want you to see a generic browser prompt, they want you to see the eCorp logo and feel warm fuzzies about their brand. This is true of HTTP `Basic-Auth` authentication, and it's also true of TLS client certificate authentication.

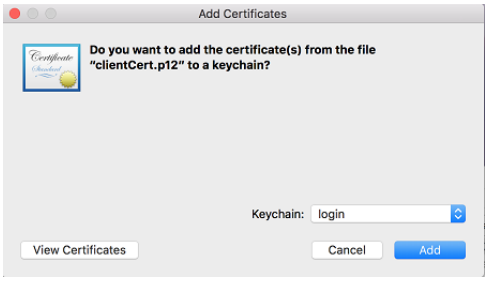
Client certificate issues

User experience

- unusual
- pre-interaction

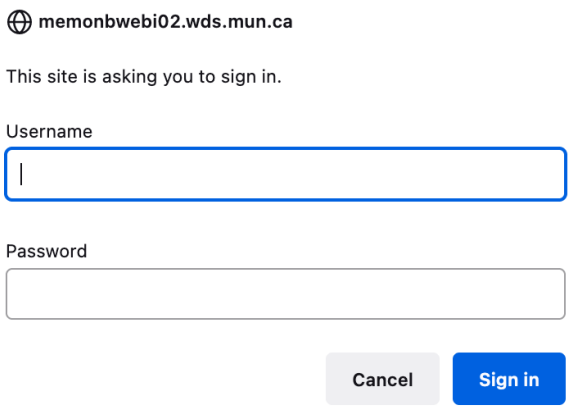
TLS termination and trust

- why believes what? why?



Source: Microsoft

It's very unusual to be prompted for a TLS client certificate. In fact, it's even unusual to see a browser-native password prompt rather than an HTML form:



One reason for this is that it doesn't give the user a chance to see the website that they're logging into. When you visit ecorp.com, they don't want you to see a generic browser prompt, they want you to see the eCorp logo and feel warm fuzzies about their brand. This is true of HTTP `Basic-Auth` authentication, and it's also true of TLS client certificate authentication.

It's very typical to have TLS connections terminated by one host which acts as a proxy for internal hosts. In that case, the internal host has _____: it just has to _____ for the user's identity.

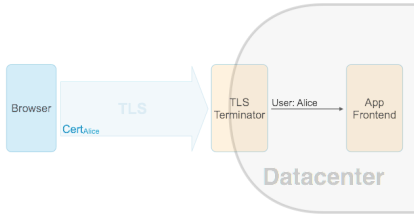
Client certificate issues

User experience

- unusual action
- pre-interaction privacy

TLS termination and trust

- why believes what? why?



Source: browserauth.net

It's very unusual to be prompted for a TLS client certificate. In fact, it's even unusual to see a browser-native password prompt rather than an HTML form:

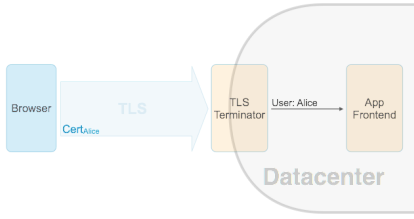
One reason for this is that it doesn't give the user a chance to see the website that they're logging into. When you visit ecorp.com, they don't want you to see a generic browser prompt, they want you to see the eCorp logo and feel warm fuzzies about their brand. This is true of HTTP `Basic-Auth` authentication, and it's also true of TLS client certificate authentication.

It's very typical to have TLS connections terminated by one host which acts as a proxy for internal hosts. In that case, the internal host has _____: it just has to _____ for the user's identity.

Client certificate issues

User experience

- unusual action
- pre-interaction privacy



Source: browserauth.net

TLS termination and trust

- why believes what? why?

Useful in *certain* circumstances

It's very unusual to be prompted for a TLS client certificate. In fact, it's even unusual to see a browser-native password prompt rather than an HTML form:

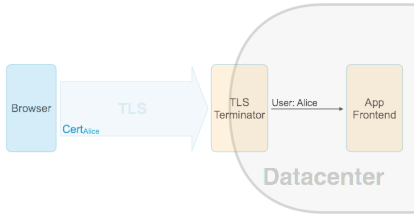
One reason for this is that it doesn't give the user a chance to see the website that they're logging into. When you visit ecorp.com, they don't want you to see a generic browser prompt, they want you to see the eCorp logo and feel warm fuzzies about their brand. This is true of HTTP `Basic-Auth` authentication, and it's also true of TLS client certificate authentication.

It's very typical to have TLS connections terminated by one host which acts as a proxy for internal hosts. In that case, the internal host has _____: it just has to _____ for the user's identity.

Client certificate issues

User experience

- unusual action
- pre-interaction privacy



Source: browserauth.net

TLS termination and trust

- why believes what? why?

Useful in *certain* circumstances (corporate, end-to-end M2M...)

It's very unusual to be prompted for a TLS client certificate. In fact, it's even unusual to see a browser-native password prompt rather than an HTML form:

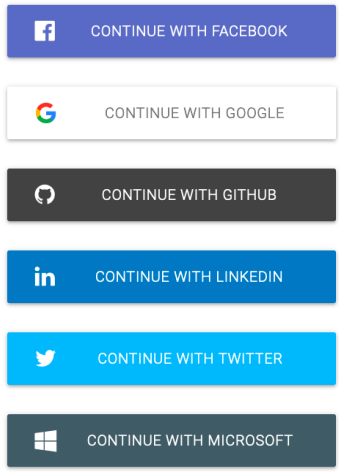
One reason for this is that it doesn't give the user a chance to see the website that they're logging into. When you visit ecorp.com, they don't want you to see a generic browser prompt, they want you to see the eCorp logo and feel warm fuzzies about their brand. This is true of HTTP `Basic-Auth` authentication, and it's also true of TLS client certificate authentication.

It's very typical to have TLS connections terminated by one host which acts as a proxy for internal hosts. In that case, the internal host has _____: it just has to _____ for the user's identity.

That said, client cert authentication *can* be useful when you can expect users to be trained to expect the prompts and respond appropriately. This can be true in corporate settings, and it's definitely

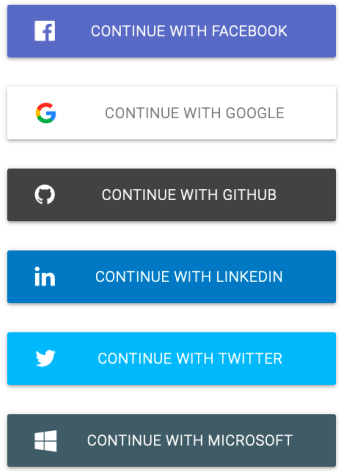
the prompts and respond appropriately. This can be true in corporate settings, and it's definitely true in M2M environments (where "training" is also known as "programming").

OAuth


CONTINUE WITH FACEBOOK
CONTINUE WITH GOOGLE
CONTINUE WITH GITHUB
CONTINUE WITH LINKEDIN
CONTINUE WITH TWITTER
CONTINUE WITH MICROSOFT

OAuth

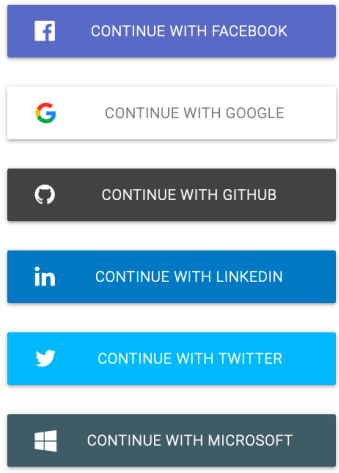
RFC ~~5849~~ 6749



OAuth

RFC ~~5849~~ 6749

Several parties:

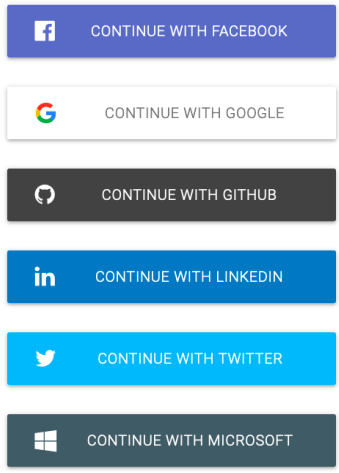


OAuth

RFC ~~5849~~ 6749

Several parties:

- user

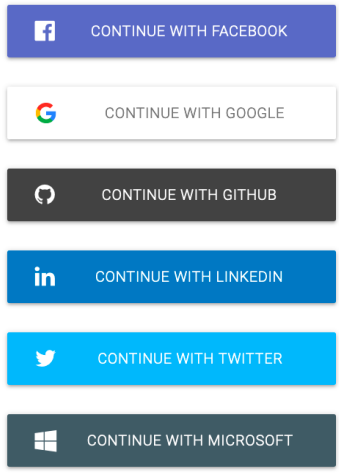


OAuth

RFC ~~5849~~ 6749

Several parties:

- user
- application developer

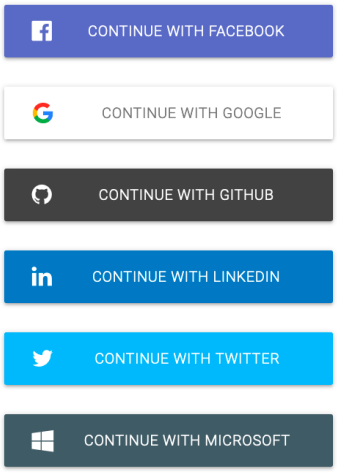


OAuth

RFC ~~5849~~ 6749

Several parties:

- user
- application developer
- authentication provider



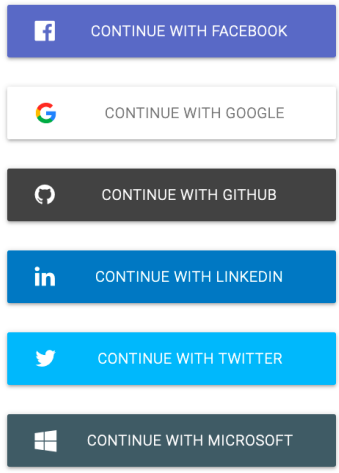
OAuth

RFC ~~5849~~ 6749

Several parties:

- user
- application developer
- authentication provider

Mutual distrust



OAuth2 protocol

Rather... flexible

OAuth2 protocol

Rather... flexible

> *OAuth 2.0 provides a rich authorization framework with well-defined security properties. However, as a rich and highly extensible framework with many optional components, on its own, this specification is likely to produce a* ***wide range of non-interoperable implementations****.*

OAuth2 protocol

Rather... flexible

> *OAuth 2.0 provides a rich authorization framework with well-defined security properties. However, as a rich and highly extensible framework with many optional components, on its own, this specification is likely to produce a* ***wide range of non-interoperable implementations****.*

Also see: OAuth 2.0 and the Road to Hell

OAuth2 protocol

Rather... flexible

> *OAuth 2.0 provides a rich authorization framework with well-defined security properties. However, as a rich and highly extensible framework with many optional components, on its own, this specification is likely to produce a* ***wide range of non-interoperable implementations****.*

Also see: OAuth 2.0 and the Road to Hell

OAuth in practice

OAuth setup

> *this specification leaves a few required components partially or fully undefined (e.g., client registration, authorization server capabilities, endpoint discovery)*

OAuth setup

> *this specification leaves a few required components partially or fully undefined (e.g., client registration, authorization server capabilities, endpoint discovery)*

Client registration

OAuth setup

> *this specification leaves a few required components partially or fully undefined (e.g., client registration, authorization server capabilities, endpoint discovery)*

Client registration

Application setup

OAuth setup

> *this specification leaves a few required components partially or fully undefined (e.g., client registration, authorization server capabilities, endpoint discovery)*

Client registration

Application setup

- register application with auth provider

OAuth setup

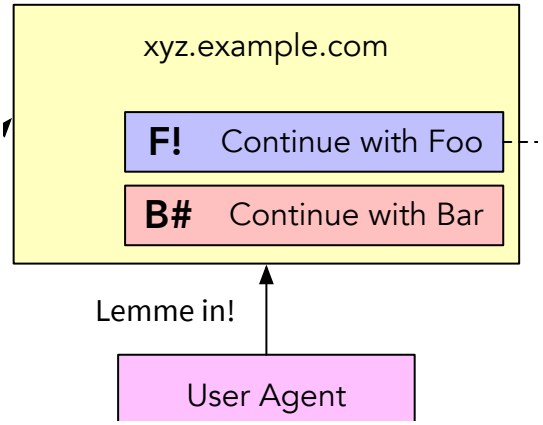
> *this specification leaves a few required components partially or fully undefined (e.g., client registration, authorization server capabilities, endpoint discovery)*

Client registration

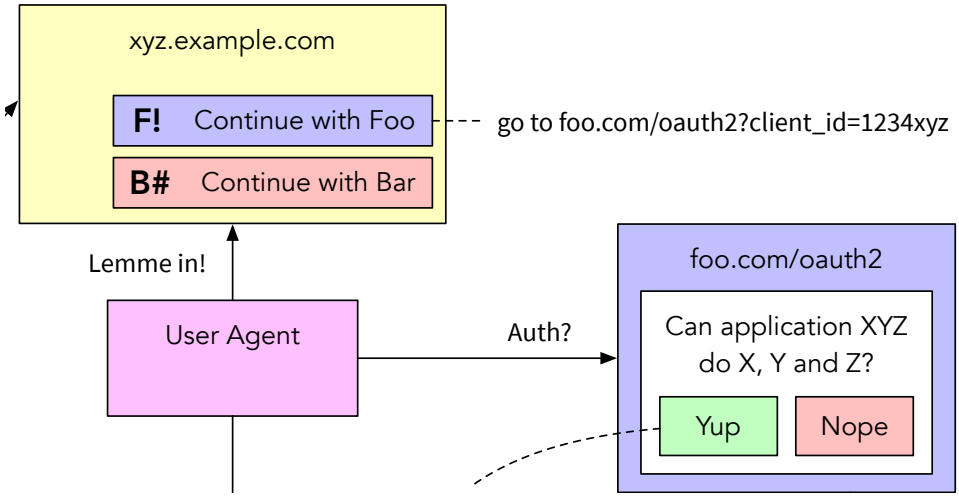
Application setup

- register application with auth provider: ID and **secret**

Login request

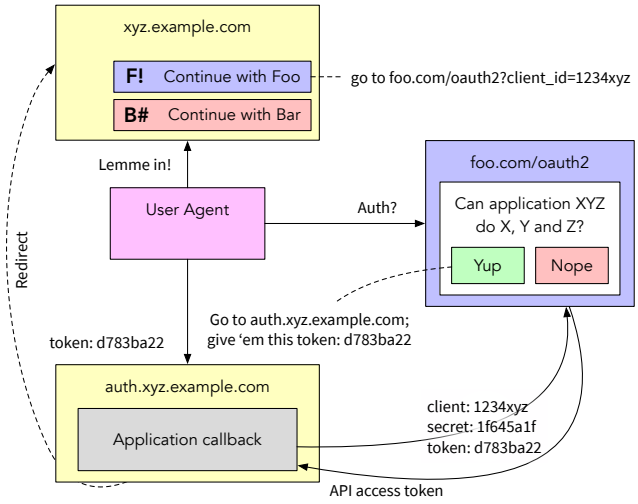


Auth redirection



Authorization code

Access token



More OAuth details

Omitted some details:

More OAuth details

Omitted some details:

- authorization *grant* types (client credentials, etc.)

The typical grant type is the *Authorization Code Grant*, described in §4.1 of RFC 6749. However, other grant types (e.g., *Implicit Grant*, which doesn't cause a back-end communication between the Client and the Authorization Server involving a Client secret hidden from the user) are also possible.

More OAuth details

Omitted some details:

- authorization *grant* types (client credentials, etc.)
- access *scope*

The typical grant type is the *Authorization Code Grant*, described in §4.1 of RFC 6749. However, other grant types (e.g., *Implicit Grant*, which doesn't cause a back-end communication between the Client and the Authorization Server involving a Client secret hidden from the user) are also possible.

An OAuth *scope* has syntax defined by the RFC (space-separated strings) with only minimal semantics (additive composition, order doesn't matter). Further semantics (i.e., what each scope means) are defined by the authorization server. For example, here are GitHub's scopes.

More OAuth details

Omitted some details:

- authorization *grant* types (client credentials, etc.)
- access *scope*

Can read RFC 6749, *but:*

Note the definition of "client" carefully!

The typical grant type is the *Authorization Code Grant*, described in §4.1 of RFC 6749. However, other grant types (e.g., *Implicit Grant*, which doesn't cause a back-end communication between the Client and the Authorization Server involving a Client secret hidden from the user) are also possible.

An OAuth *scope* has syntax defined by the RFC (space-separated strings) with only minimal semantics (additive composition, order doesn't matter). Further semantics (i.e., what each scope means) are defined by the authorization server. For example, here are GitHub's scopes.

As per §1.1 of RFC 6749, the "client" doesn't refer to the end user: the user is the "resource owner". Instead, the "client" is an application that makes requests. This is typically the user's "user agent" (browser), but it can be another application, like a mail client.

Cookies

Cookies

Session cookies

Cookies

Session cookies

- crypto?

Cookies

Session cookies

- crypto?
- *sidejacking*

Sidejacking refers to the the practice of sniffing authentication cookies over a plain-text connection and then using them to impersonate a user. This was a dangerous attack back in the dark days of the early 2010s, when it was common for even large websites to do authentication over HTTPS but then allow regular browing over HTTP. More and more, however, sites are using HTTPS all the time, preventing this kind of attack.

Cookies

Session cookies

- crypto?
- *sidejacking*

Login cookies

Sidejacking refers to the the practice of sniffing authentication cookies over a plain-text connection and then using them to impersonate a user. This was a dangerous attack back in the dark days of the early 2010s, when it was common for even large websites to do authentication over HTTPS but then allow regular browing over HTTP. More and more, however, sites are using HTTPS all the time, preventing this kind of attack.

Cookies

Session cookies

- crypto?
- *sidejacking*

Login cookies

- crypto?

Sidejacking refers to the the practice of sniffing authentication cookies over a plain-text connection and then using them to impersonate a user. This was a dangerous attack back in the dark days of the early 2010s, when it was common for even large websites to do authentication over HTTPS but then allow regular browing over HTTP. More and more, however, sites are using HTTPS all the time, preventing this kind of attack.

Cookies

Session cookies

- crypto?
- *sidejacking*

Login cookies

- crypto?
- "Remember me on this computer"

Sidejacking refers to the the practice of sniffing authentication cookies over a plain-text connection and then using them to impersonate a user. This was a dangerous attack back in the dark days of the early 2010s, when it was common for even large websites to do authentication over HTTPS but then allow regular browing over HTTP. More and more, however, sites are using HTTPS all the time, preventing this kind of attack.

Authentication signals

Many *fallible* ways to build authentication comfort:

As we've seen throughout the course, security mechanisms don't have to be perfect in order to be beneficial, as long as they aren't the primary line of defence. An authentication *signal* doesn't provide authentication by itself, but like a Customs officer who picks up on a traveler acting nervous, it can provide a prompt to dig deeper into the primary sources.

Authentication signals

Many *fallible* ways to build authentication comfort:

- IP address

As we've seen throughout the course, security mechanisms don't have to be perfect in order to be beneficial, as long as they aren't the primary line of defence. An authentication *signal* doesn't provide authentication by itself, but like a Customs officer who picks up on a traveler acting nervous, it can provide a prompt to dig deeper into the primary sources.

Authentication signals

Many *fallible* ways to build authentication comfort:

- IP address
- Browser fingerprint

As we've seen throughout the course, security mechanisms don't have to be perfect in order to be beneficial, as long as they aren't the primary line of defence. An authentication *signal* doesn't provide authentication by itself, but like a Customs officer who picks up on a traveler acting nervous, it can provide a prompt to dig deeper into the primary sources.

Authentication signals

Many *fallible* ways to build authentication comfort:

- IP address
- Browser fingerprint
- Time of day

As we've seen throughout the course, security mechanisms don't have to be perfect in order to be beneficial, as long as they aren't the primary line of defence. An authentication *signal* doesn't provide authentication by itself, but like a Customs officer who picks up on a traveler acting nervous, it can provide a prompt to dig deeper into the primary sources.

Authentication signals

Many *fallible* ways to build authentication comfort:

- IP address
- Browser fingerprint
- Time of day
- User behaviour (e.g., queries)

As we've seen throughout the course, security mechanisms don't have to be perfect in order to be beneficial, as long as they aren't the primary line of defence. An authentication *signal* doesn't provide authentication by itself, but like a Customs officer who picks up on a traveler acting nervous, it can provide a prompt to dig deeper into the primary sources.

Authentication signals

Many *fallible* ways to build authentication comfort:

- IP address
- Browser fingerprint
- Time of day
- User behaviour (e.g., queries)

Maybe let's re-authenticate?

As we've seen throughout the course, security mechanisms don't have to be perfect in order to be beneficial, as long as they aren't the primary line of defence. An authentication *signal* doesn't provide authentication by itself, but like a Customs officer who picks up on a traveler acting nervous, it can provide a prompt to dig deeper into the primary sources.

Summary

Authentication factors

TLS client certificates

OAuth

Cookies

Authentication signals

The End.