IDENTITY AND ACCESS MANAGEMENT:
Past/present/future, SAML, OAuth, FIDO, OIDC, other acronyms, and emerging trends

Brian Campbell
Distinguished Engineer
Ping Identity
@___b_c
About Me:

- Distinguished Engineer at Ping Identity
- Likes to take pictures
Welcome to #RSAC!
I am going to talk about IAM

Identity and Access Management
let the right people access what they need
keep the wrong people out
1961: Password Invented
Okay, passwords are ancient

But first known computer use was in ‘61
  - at MIT for the Compatible Time-Sharing System
    - each user had a private set of files and allotment of computing time

Even back then IAM was about the right people having access to the right things at the right time

System defeated just one year later
  - request to print the password file offline
Sixteen years later I was born

(not actually me)
And I’m a little hazy on what happened in that time.
Twenty-Some Years Later

- The World Wide Web is Now a Thing
- HTTP Basic Authentication
  - Per application credentials
  - Centralized LDAP
  - credentials sent & checked on every request
- HTML form based login
  - Cookie based session established from login
  - Typically opaque value referencing server side memory
- Around this time I’d write my first single sign-on system...
Which had some serious problems...

(blindly trusting a user id value in a site-wide cookie, what could possibly go wrong?)
Luckily, competent people were also working on it

- Web Access Management (WAM) Products/Solutions
  - Single sign-on, authorization policy, and authentication management
  - Web server agent (but sometimes also reverse proxies)
  - Domain-wide cookie (but secured unlike mine)
  - Centralized policy server
  - Typically deployed in
    - Large consumer web sites
    - Enterprise applications behind the firewall
  - Cross-domain solutions existed but proprietary & non-interoperable
Cross Domain Standardization Efforts Also Underway

SAML 1.0, 1.1 & 2.0

ID-FF 1.0, 1.1 & 1.2
A few years later sees the rise of SaaS (as we know it now) accelerating the need for cross-domain single sign-on.
It's a SaaS world after all
It's a SaaS world after all
It's a SaaS world after all
How does that make you feel?
- Too many damn Passwords
- Inconsistent policies
- Stronger authentication, if any, is per SaaS
SAML Single Sign-On to SaaS
SAML Single Sign-On to SaaS

SSO Server

User Authenticates

Directory

USERNAME
bcampbell

PASSWORD

Remember my username

Sign On

Change Password?

vendingforce

Workplace 24/7

MeetingEx
SAML Single Sign-On to SaaS

SSO Server

User Authenticates →

Directory

Authenticate on Phone

Change Device

Change Password?

vendingforce

Workplace 24/7

MeetingEx
SAML Single Sign-On to SaaS

SSO Server

SAML Single Sign-On to SaaS

SAML Assertion → Session Cookie

SAML Assertion ← Session Cookie

Directory

vendingforce

Workplace 24/7

MeetingEx
SAML Single Sign-On to SaaS

SSO Server

Authn Request & Session Cookie

vendingforce

Workplace 24/7

MeetingEx

← Authn Request
SAML Single Sign-On to SaaS

SSO Server

SAML Assertion →

Directory

SAML Assertion →

Session Cookie

vendigforce

Workplace 24/7

MeetingEx
SAML Single Sign-On to SaaS

et cetera, et cetera, et cetera, etc.
SAML: XML standard for exchanging security & identity information

<assertion ID="y2bvAdFrnRNvm103yjiimgjhw7" IssueInstant="2016-12-05T21:38:44.771Z" Version="2.0" xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
  <issuer>https://pongidentity.com</issuer>
  <signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#"/>
  <canonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
  <signatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsasha256"/>
  <reference URI="#y2bvAdFrnRNvm103yjiimgjhw7"/>
  <signatureValue>gZbkpGU[...omitted...o2riMFGnTraY=</signatureValue>
  <subject>
    <nameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified">bcampbell</nameID>
    <subjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
      <subjectConfirmationData Recipient="https://workplace247.com/ACS" NotOnOrAfter="2016-12-05T21:48:44.771Z"/>
    </subjectConfirmation>
  </subject>
  <conditions NotBefore="2016-12-05T21:33:44.771Z" NotOnOrAfter="2016-12-05T21:48:44.771Z">
    <audienceRestriction><audience>urn:federation:workplace-24-7</audience></audienceRestriction>
  </conditions>
  <authnStatement SessionIndex="y2bvAdFrnRNvm103yjiimgjhw7" AuthnInstant="2016-12-05T21:27:35.000Z">
    <authnContext>
      <authnContextClassRef>urn:oasis:names:tc:SAML:2.0:ac:classes:Password</authnContextClassRef>
    </authnContext>
  </authnStatement>
  <attributeStatement xsi:instance="http://www.w3.org/2001/XMLSchema-instance">
    <attribute Name="fname" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic">
      <value xsi:type="xs:string">Brian</value>
    </attribute>
    <attribute Name="lname" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic">
      <value xsi:type="xs:string">Campbell</value>
    </attribute>
    <attribute Name="email" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic">
      <value xsi:type="xs:string">bcampbell@pongidentity.com</value>
    </attribute>
  </attributeStatement>
</assertion>
OAuth Drivers: Password Sharing is Bad

Other sites asks YOU for your <redacted> password so it can access your <redacted> stuff.
OAuth Drivers: SOAP -> REST & JSON

but there were no comparable authentication & authorization standards to WS-*
OAuth 2.0 In A Nutshell

Client → Authorization Server → Resource Server

1. Client requests an access token from the Authorization Server.
3. Client exchanges the authorization code for an access token from the Authorization Server.
4. Client uses the access token to access protected resources on the Resource Server.

Diagram:
- Client
- Authorization Server
- Resource Server
- Get an access token
- Use an access token
OpenID Connect: SSO built on OAuth 2.0

- “OpenID Connect is a simple identity layer on top of the OAuth 2.0 protocol.”
- Simple is in the eye of the beholder
  - But complexity burden largely shifted to the identity provider
- Adoption in both employee and consumer use-cases
- Adds a lot to OAuth
  - But the main thing is the JSON Web Token (JWT) based ID Token

(Client) → Authorization Server
Get an access token + an ID Token (JWT)
(Client) ← Resource Server
Use an access token
The JWT

```
eyJraWQiOiI1IiwiYWxnIjoiRVMyNTYifQ.eyJpc3MiOiJodHRwczpcL1wvaWRwLmV4YW1wbGUuY29tIiwKImV4cCI6MTM1NzI1NTc4OCwKImF1ZCI6Imh0dHBzOlwvXC9zcC5leGFtcGxlLm9yZyIsCiJqdGkiOiJ0bVl2WVZVMng4THZONzJCNVFfRWFjSC5fNUEiLAoiYWNYIjoiMiIsCiJzdWIiOiJCCmlhbiJ9.SbPJlx_JSRM1wuioY0SvfykKWK_yK4LO0BKBiESHu0GUGwikgC8iPrv8qnVbIK1aljVbVMXcbgYnZixZJ5U0Arg
```

The Header

```
{"kid":"5","alg":"ES256"}
```

The Payload

```
{"iss":"https:\/\/idp.example.com","exp":1357255788,"aud":"https:\/\/sp.example.org","jti":"tmYvYVU2x8LwN72B5Q_EacH._5A","acr":"2","sub":"Brian"}
```

The Signature
it’s not the size of your token...

eyJraWQiOiI1IiwiYmxvY2FsZXRlIjoiQ192IiwiYWxnIjoiRVMyNTYifQ.eyJpc3MiOiJodHRwczpcL1wvaWRwLmV4YW1wbGUuY29tIiwiYmV4cCI6MjMwMDQ4MjA5MywiY2hhbmsiOiI0MDE4MTk2MzkiLCJpZCI6MjUyODY3MCwiaG9yaVRlcCI6IjI0MDE4MTk2MCIiLCJzdWIiOiJiZyIsImRldmVyaWFsaW5lRGVzY3JpcHRpb24iOjEyNDRiOTM3NjgyOTB6LCJoZWlnaHQiOiJ1c2VyIiwiZXhwaXJlRnVpZCI6IjI0MDE4MTk2MCIiLCJzb3VyY2VSZWFkeS1uYW1lIjoiQWZ0Iiwid2VicGFjaXR5IjoicmVzaXplIn0.

<Assertion Version="2.0" IssueInstant="2013-01-03T23:34:38.546Z" ID="oPm.DxOqT3Zi83IwuVr3x83xlR"
xmlns="urn:oasis:names:tc:SAML:2.0:assertion"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  <Issuer>https://idp.example.com</Issuer>
  <ds:Signature>
    <ds:SignedInfo>
      <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
      <ds:SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha256"/>
      <ds:Reference URI="#oPm.DxOqT3Zi83IwuVr3x83xlR">
        <ds:Transforms>
          <ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
          <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
        </ds:Transforms>
        <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
        <ds:DigestValue>8JT03jJlsqBxHstxmDhs2z1CPsgMkMTC11IK9g7e0o==</ds:DigestValue>
      </ds:Reference>
    </ds:SignedInfo>
    <ds:SignatureValue>SAXf8eCmTjuhV742blyvLvvUmZJ+Tqi3G3eMsRDUQU8RnNSspZzNJ8MOUwffkT6kvARDXeVzob5p08jsb99UJQ==</ds:SignatureValue>
  </ds:Signature>
  <Subject>
    <NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified">Brian</NameID>
    <SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
      <SubjectConfirmationData NotOnOrAfter="2013-01-03T23:39:38.552Z" Recipient="https://sp.example.org"/>
    </SubjectConfirmation>
  </Subject>
    <AudienceRestriction />
  </Conditions>
  <AuthnStatement AuthnInstant="2013-01-03T23:34:38.4832" SessionIndex="oPm.DxOqT3Zi83IwuVr3x83xlR">
    <AuthnContext>
      <AuthnContextClassRef>2</AuthnContextClassRef>
    </AuthnContext>
  </AuthnStatement>
</Assertion>
...it’s how you use it

- Simpler = Better
- Web safe encoding w/ no canonicalization (Because canonicalization is a four letter word*)
- Improved Interoperability & Security
  - Mostly been true but has its critics...
- Eliminates entire classes of attacks (vs. XML DSIG)
  - XSLT Transform DOS, Remote Code Execution, and Bypass
  - C14N Hash Truncation
  - Entity Expansion Attacks
  - XPath Transform DOS and Bypass
  - External Reference DOS
  - Signature Wrapping Attacks
  - Bypass from inconsistent treatment of XML comments in c14n and XML APIs [new!]

* especially when you spell it c14n
Analysts* Predict 5.43 Zillion Mobile Devices by 2021

* Might have been me
OAuth 2.0 used for sign-on with native mobile applications
OAuth 2.0 for Native Apps

1. Request authorization + PKCE
2. User authentication & approval
3. Callback to custom scheme URI
4. Exchange code + PKCE for tokens
5. Access protected APIs with token
Enables Federated and Multi-factor Sign-on

Leveraging existing and future investment in web based authentication
• Standardized Online Authentication Using Public Key Cryptography
• PKI without the I
• UAF & U2F
Fast IDentity Online U2F

- Strong cryptographic 2nd factor option for end user security
- U2F device: USB, NFC, Bluetooth LE, on-board machine/mobile
- Registration of client generated site-specific public key
- Authentication by signing a challenge
A New Home for FIDO

- From the FIDO Alliance to the Web Authentication Working Group in the W3C
  - Defining a client-side API providing strong authentication functionality to web applications
  - With the FIDO 2.0 APIs as input
What’s In Your Pocket?

Phone becoming a nearly ubiquitous “something you have”
While standards make having fewer hard token(s) feasible
Biometrics

Used as device local authentication to unlock a key used in remote authentication
Token Binding

• Enables a long-lived binding to browser generated public-private key pair used to sign TLS exported keying material and sent as an HTTP header
• Bind to cookies, SSO tokens, OAuth tokens
Are we done yet?

- IAM: Seamlessly enabling the right people to have access to the right resources at the right time
  - Federated single sign-on to SaaS & organizational applications deployed wherever (or “social” login to consumer apps)
  - Stronger user authentication with less frequent direct user interaction
  - Stronger session and SSO tokens bound to keys on the device
Thanks!

You’ve been watching:

IDENTITY AND ACCESS MANAGEMENT:
Past/present/future, SAML, OAuth, FIDO, OIDC, other acronyms, and emerging trends

Brian Campbell
Distinguished Engineer
Ping Identity
@__b_c