moz://a

Developer Roadshow 2017

Galvanize, Denver, CO | July 27, 2017

Hello Denver!

Overview

We're Mozilla, the proudly non-profit champions of the Internet, helping to keep it healthy, open and accessible to all.

Tweet at Us!

#mozilla

#DevRoadshow

TAKE 3 MIN to Tell Us What you think!

And be entered to win sweet swag.

mzl.la/devsurvey

Code of Conduct

A primary goal of Mozilla's Developer Roadshow (Roadshow) is to be inclusive to the largest number of participants, with the most varied and diverse backgrounds possible. As such, we are committed to providing a friendly, safe and welcoming environment for all, regardless of gender, sexual orientation, ability, age, ethnicity, socioeconomic status, and religion (or lack thereof).

This Code of Conduct outlines our expectations for all those who participate in our conference community, as well as the consequences for unacceptable behavior.
We invite all those who participate in the Roadshow to help us create safe and positive experiences for everyone.

Please find the full text here: https://mzl.la/devroadshowcoc

And contact Sandra Persing (sandra@mozilla.com; @sandrapersing) for any issues, questions, concerns.

Me.

I am Michael Van Kleeck

I work on Enterprise Architecture

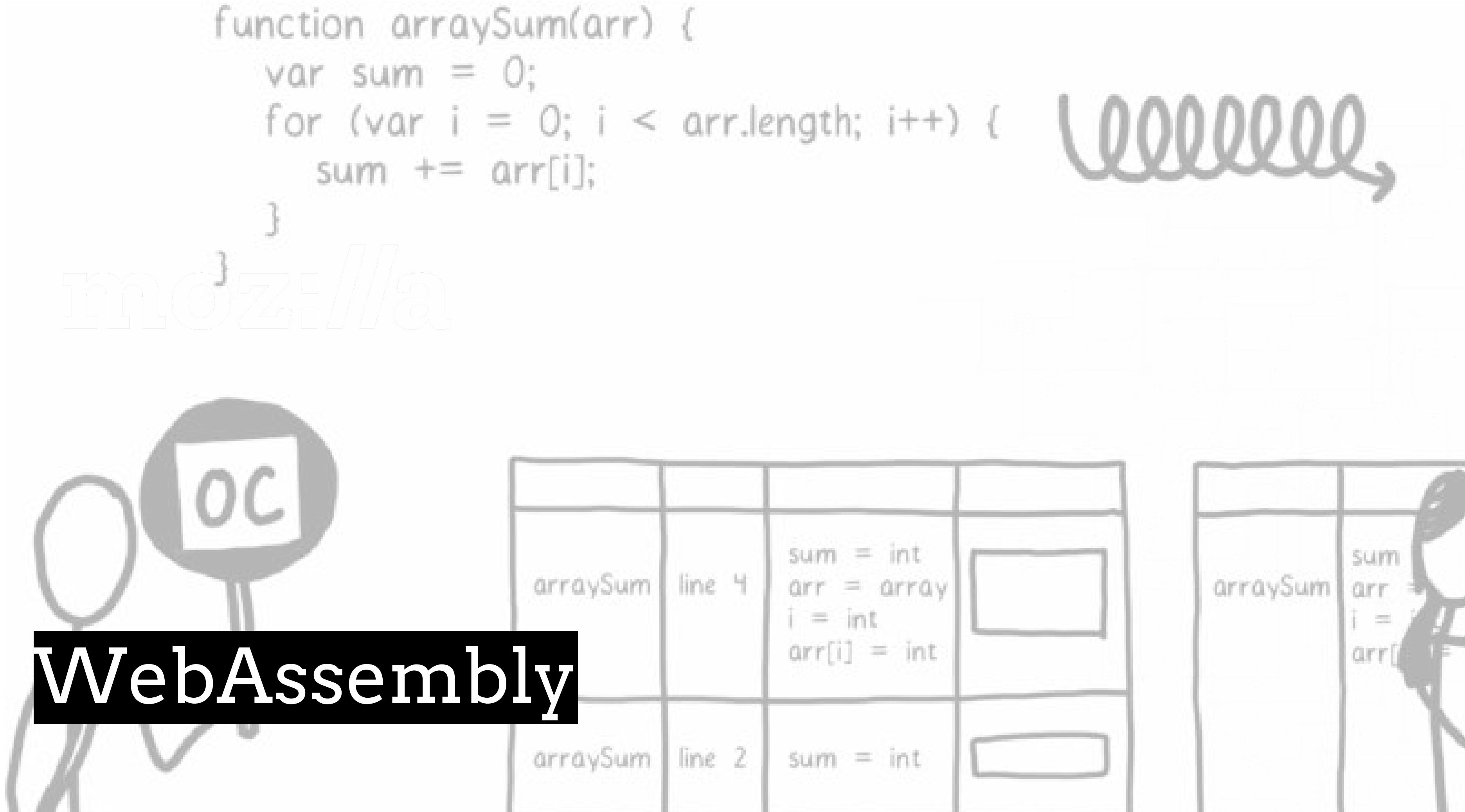
Twitter @michaelvkpdx

Major Things to Look Forward to in 2017

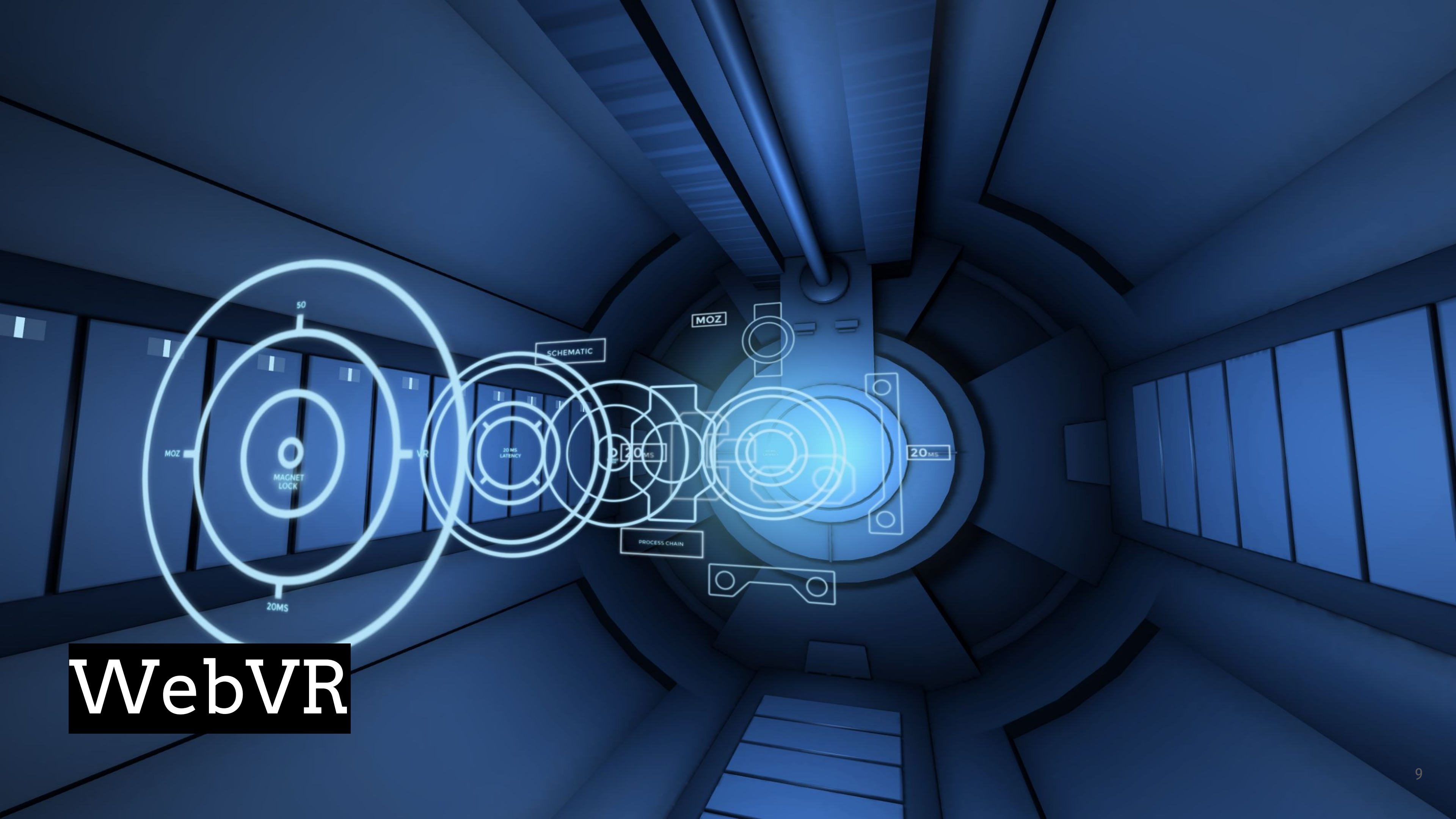
New Web Standards

1. WebAssembly
2. WebGL 2
3. WebVR
4. CSS Grid

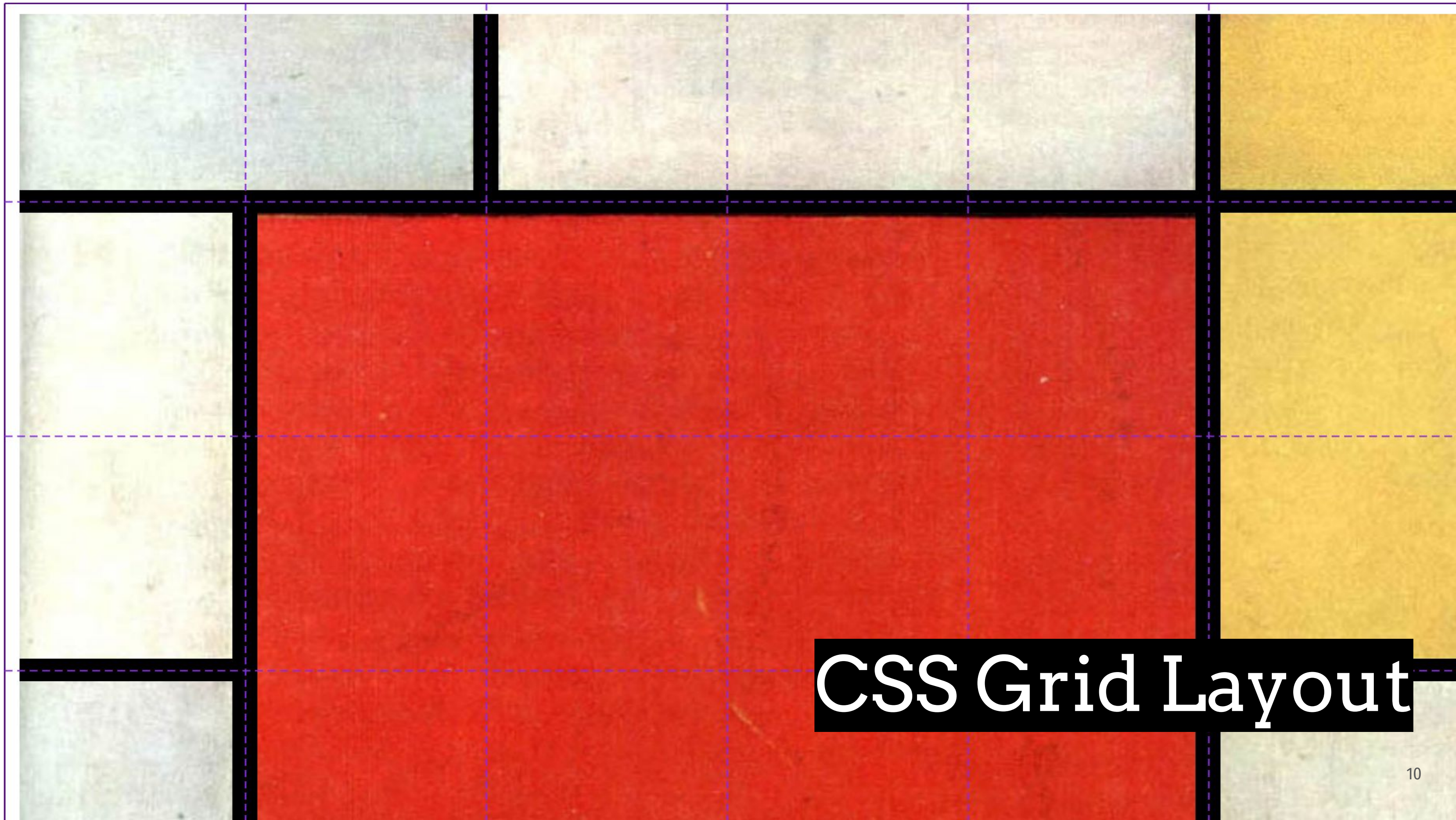
moz://a



WebAssembly


50
MOZ
VR
MAGNET LOCK
20MS
SCHEMATIC
20 MS LATENCY
MOZ
20MS
20MS
PROCESS CHAIN

WebVR



CSS Grid Layout

Major Things to Look Forward to in 2017

New Web Standards

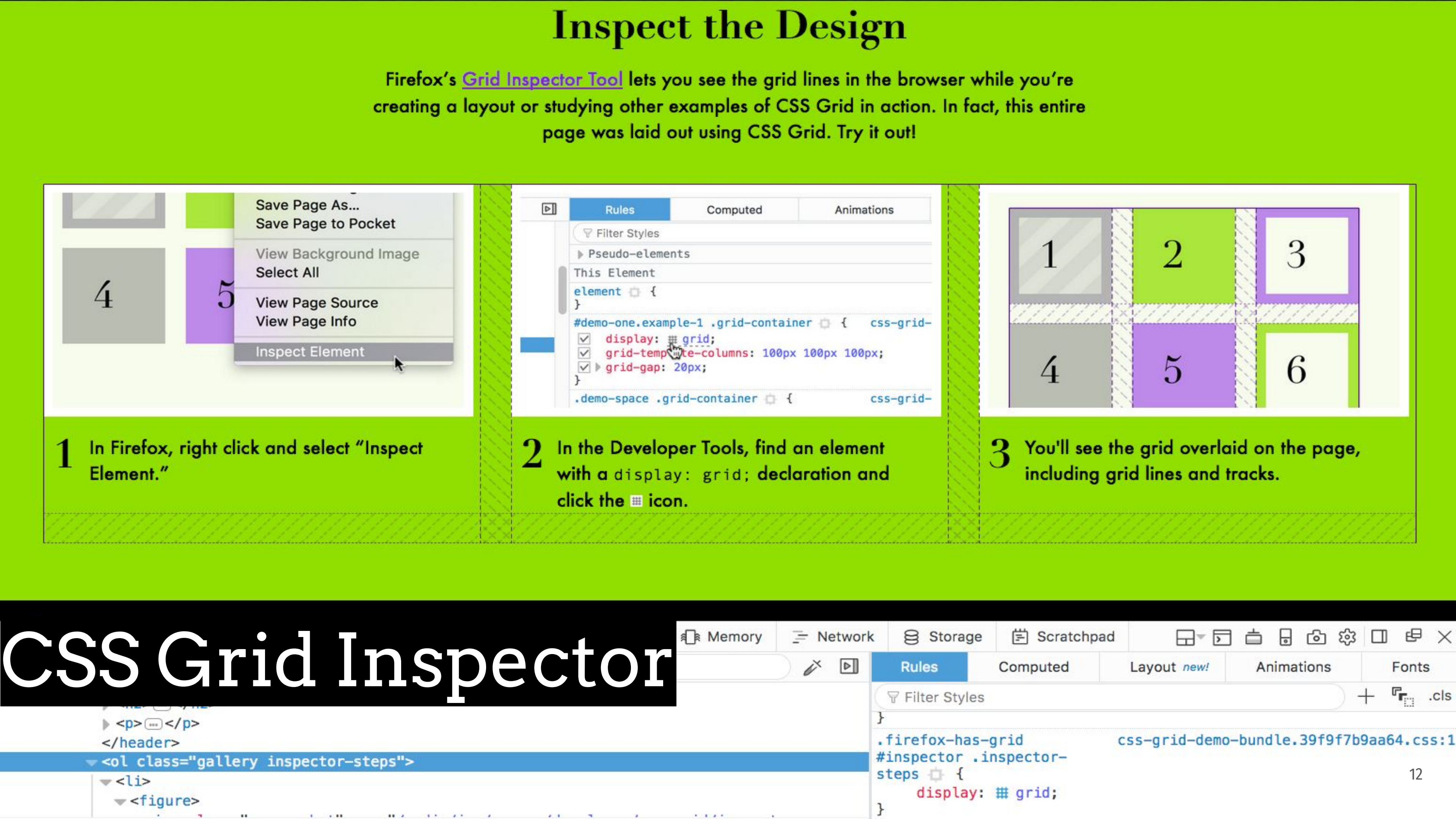
1. WebAssembly
2. WebGL 2
3. WebVR
4. CSS Grid

DevTools

1. Rewriting the DevTools into standard HTML/CSS/JS
2. Hosting the DevTools on GitHub as individual add-ons, allowing faster updates and easier outside contributions
3. CSS Grid Inspector

moz://a

CSS Grid Inspector



Inspect the Design

Firefox's Grid Inspector Tool lets you see the grid lines in the browser while you're creating a layout or studying other examples of CSS Grid in action. In fact, this entire page was laid out using CSS Grid. Try it out!

1 In Firefox, right click and select "Inspect Element."

```
element {
}
#demo-one.example-1 .grid-container {     css-grid-
  display: grid;
  grid-template-columns: 100px 100px 100px;
  grid-gap: 20px;
}
.demo-space .grid-container {     css-grid-
```

2 In the Developer Tools, find an element with a `display: grid;` declaration and click the ▦ icon.

3 You'll see the grid overlaid on the page, including grid lines and tracks.

```
<p>…</p>
</header>
<ol class="gallery inspector-steps">
  <li>
    <figure>
```

```
}
.firefox-has-grid          css-grid-demo-bundle.39f9f7b9aa64.css:1
#inspector .inspector-
steps {
    display: grid;
}
```

Major Things to Look Forward to in 2017

New Web Standards

1. WebAssembly
2. WebGL 2
3. WebVR
4. CSS Grid

DevTools

1. Rewriting the DevTools into standard HTML/CSS/JS
2. Hosting the DevTools on GitHub as individual add-ons, allowing faster updates and easier outside contributions
3. CSS Grid Inspector

Performance

1. Electrolysis (e10s)
2. Multiple content processes (e10s-multi)
3. Project Quantum (announcement)

Major Things to Look Forward to in 2017

Privacy + Security

1. The Tor Uplift
2. Strong sandboxing on all platforms
3. Flash “Ask to Activate”

Firefox Features

1. Pocket Integration
2. Activity Stream graduating from Test Pilot
3. More Test Pilot experiments
4. Container Tabs
5. Eliminating the Aurora release channel so features can get from Nightly to Release more quickly

Web Extensions

1. Standardizing add-on APIs between Firefox, Chrome, Edge, and Opera
2. Supporting the devtools.* APIs
3. Supporting the storage.sync API
4. New Firefox-specific APIs for theming the browser
5. Finishing Chrome-compat and landing more Firefox-specific APIs

Speakers

Michael Van Kleeck, Enterprise Architect:

Cloud Computing, SaaS, and Security

Chuck Harmston, Sr. Engineer- Firefox Test Pilot

Contributing to Open Source

moz://a

moz://a

moz://a

Cloud Computing, SaaS, and Security

Michael Van Kleeck
Mozilla Enterprise Solutions Architect

Cloud Computing, SaaS, Security- Overview

After this presentation, audience members will be able to:

- **Differentiate between monolithic and microservice architectures**
- **Describe what SaaS is and how it is used**
- **Compare VM and Container-based cloud architectures**
- **Talk about how to secure SaaS-driven applications using OIDC, and use JSON Web Tokens for access***

**Time permitting, some coding required*

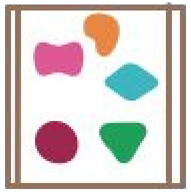
Application Architectures- a brief history

- **Mainframes and dumb terminals**
- **Client + Server**
- **Model-View-Controller**
- **Service-Oriented Architecture (Microservices)**

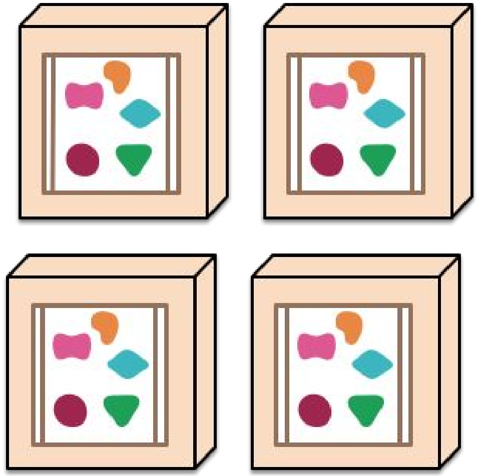
The web supports all these models. HTTP is just a protocol.

Monoliths- The Old Way, The Old New Way

A monolithic application puts all its functionality into a single process...



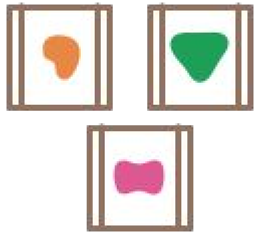
... and scales by replicating the monolith on multiple servers



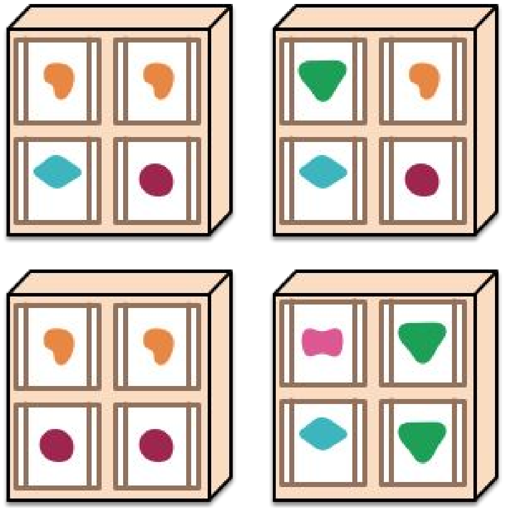
Images © Martin Fowler, *Microservices*

Microservices- The New, New Way

A microservices architecture puts each element of functionality into a separate service...



... and scales by distributing these services across servers, replicating as needed.



Images © Martin Fowler, *Microservices*

Microservices are….

- **Coarse-grained (the details are in the data)**
- **Single business function**
- **Atomic (self-contained, with rollback)**
- **Stateless**
- **Independently secure**

Examples: Amazon uses 100-150 Microservices to build a page.

Netflix runs hundreds of Microservices!

moz://a

Software As A Service (SaaS)

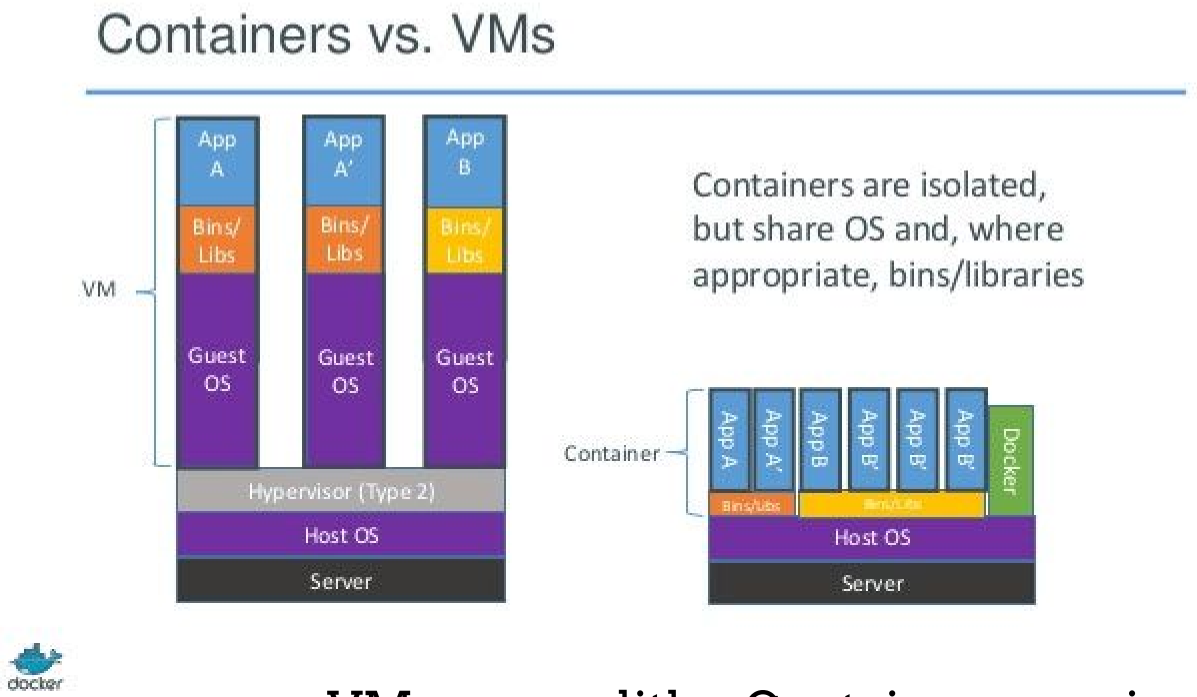
- **SaaS can be microservices- or larger services**
- **Applications combine services to implement a complete business process**
- **Independent of underlying infrastructure**
- **Widely varying in quality (buyer beware!)**

Cloud Computing

"Cloud computing is a model for enabling **ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources** (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction"

- *The NIST Definition of Cloud Computing*

Cloud Architectures- Virtual Machines vs. Containers



Containers vs. VMs

Containers are isolated, but share OS and, where appropriate, bins/libraries

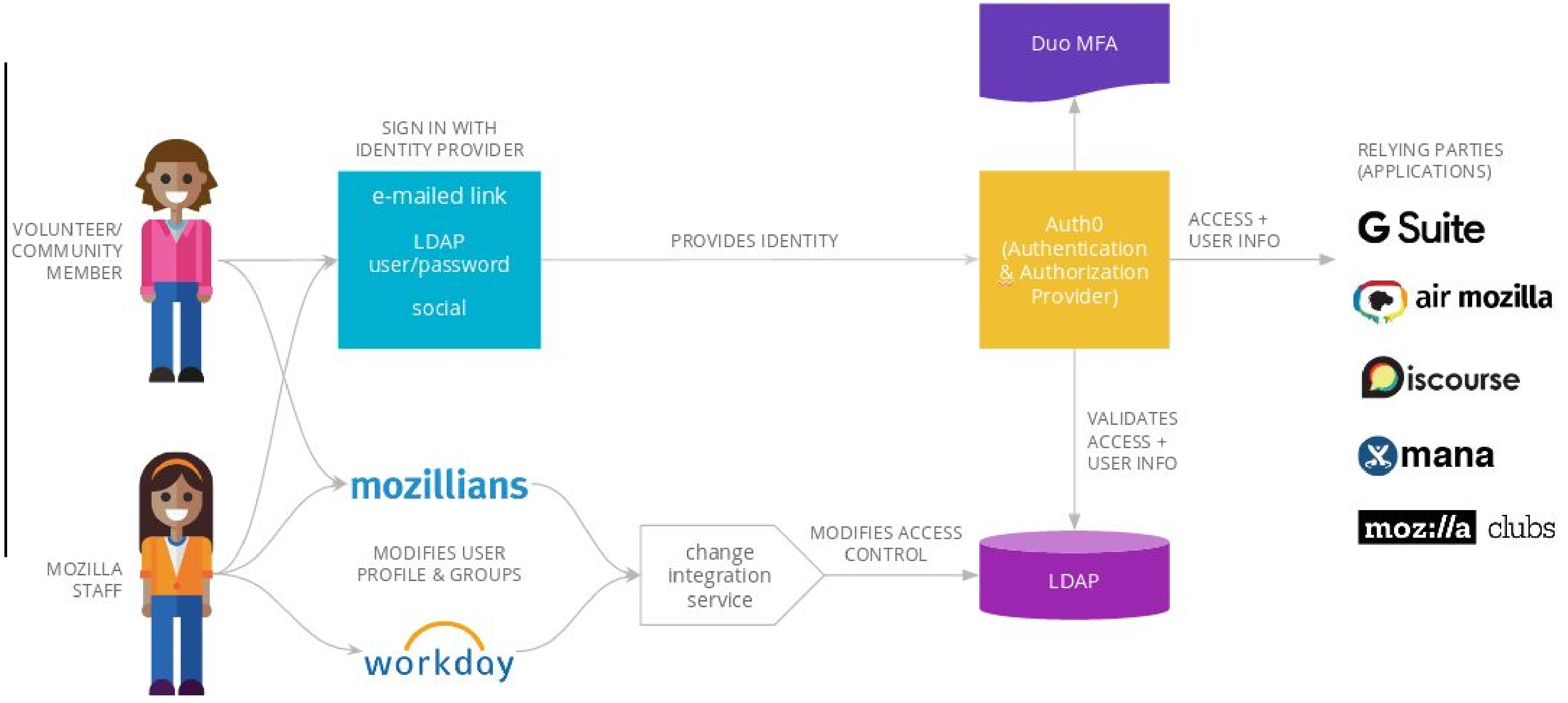
VMs ~= monliths, Containers ~= microservices

Taken from Containers vs VMs: Which is better in the Data Center? courtesy of Docker Inc. and RightScale Inc.

Software As A Service- Security

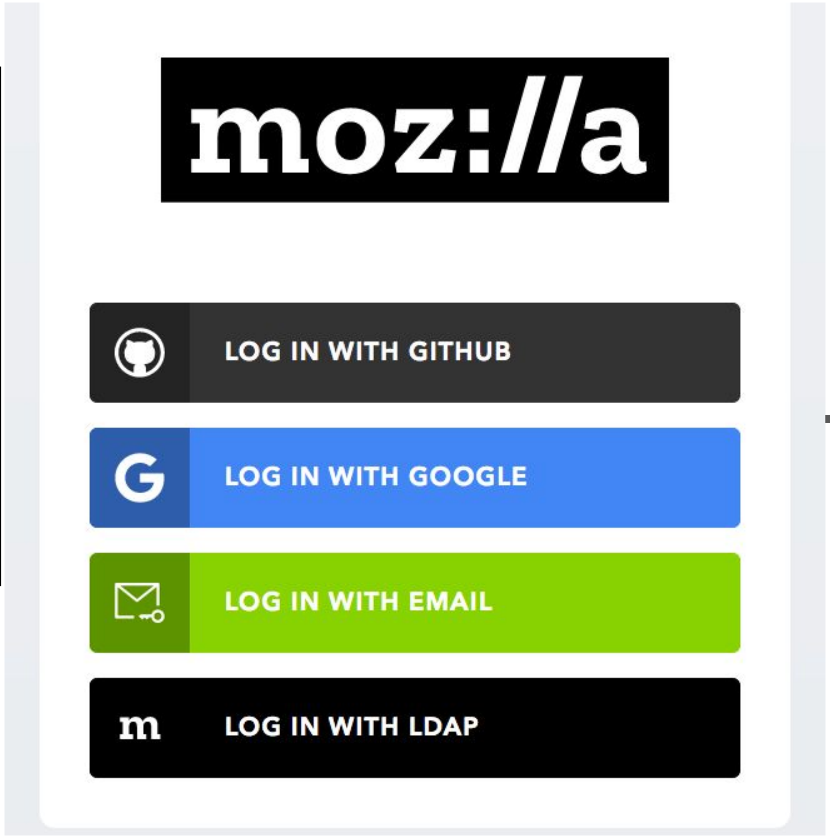
- Services require independent authentication
- Users don’t want an account with every service
- Authentication and Authorization are core differentiators for organizations
- OAuth and OIDC are the standards

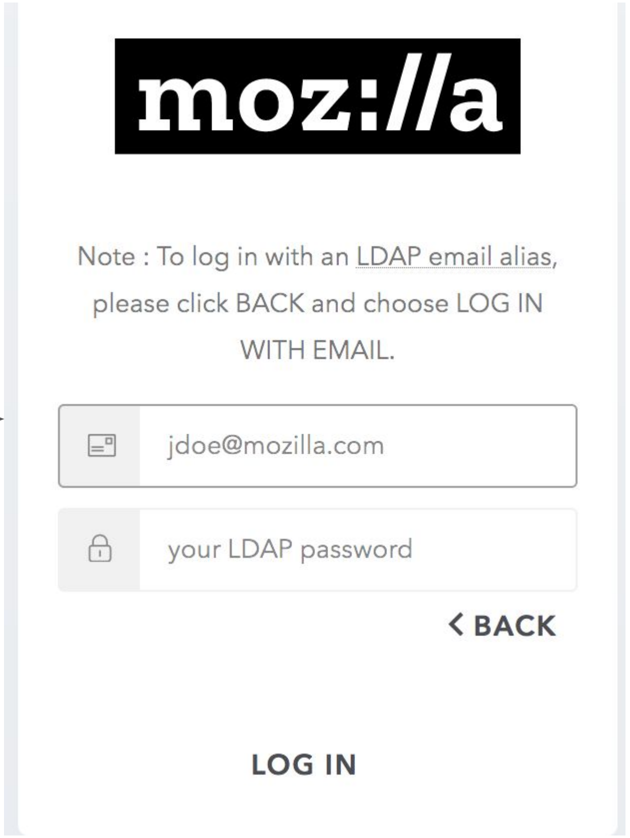
Example- Mozilla Identity and Access Management


Duo MFA
SIGN IN WITH IDENTITY PROVIDER
e-mailed link
LDAP user/password
social
RELYING PARTIES (APPLICATIONS)
VOLUNTEER/ COMMUNITY MEMBER
PROVIDES IDENTITY
Auth0 (Authentication & Authorization Provider)
ACCESS + USER INFO
G Suite
air mozilla
Discourse
mana
moz://a clubs
VALIDATES ACCESS + USER INFO
mozillians
MOZILLA STAFF
MODIFIES USER PROFILE & GROUPS
change integration service
MODIFIES ACCESS CONTROL
LDAP
workday

Note: High-security services have extra checks on access control modification

Identity and Access Management- Lock screen





Note : To log in with an LDAP email alias, please click BACK and choose LOG IN WITH EMAIL.

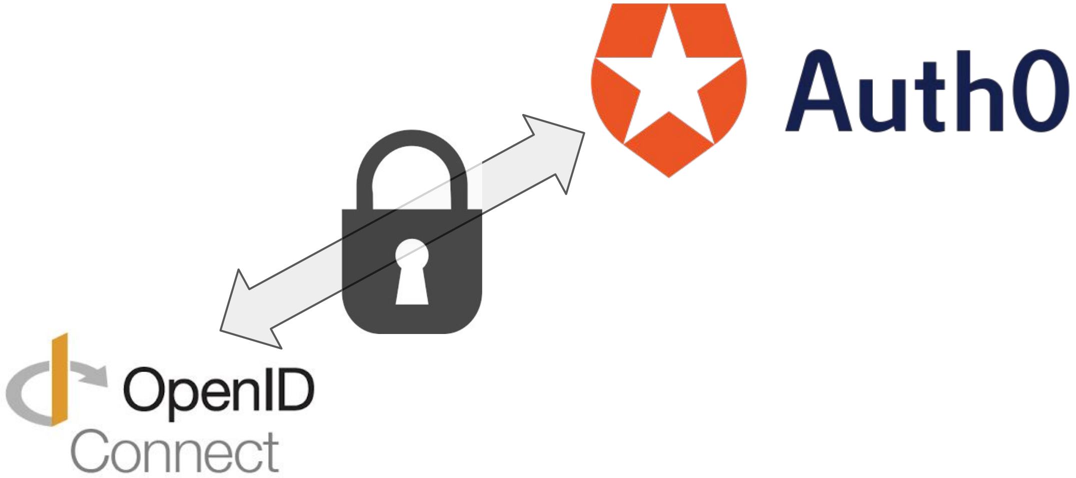
jdoe@mozilla.com

your LDAP password

< BACK

LOG IN

Securing SaaS Using OIDC

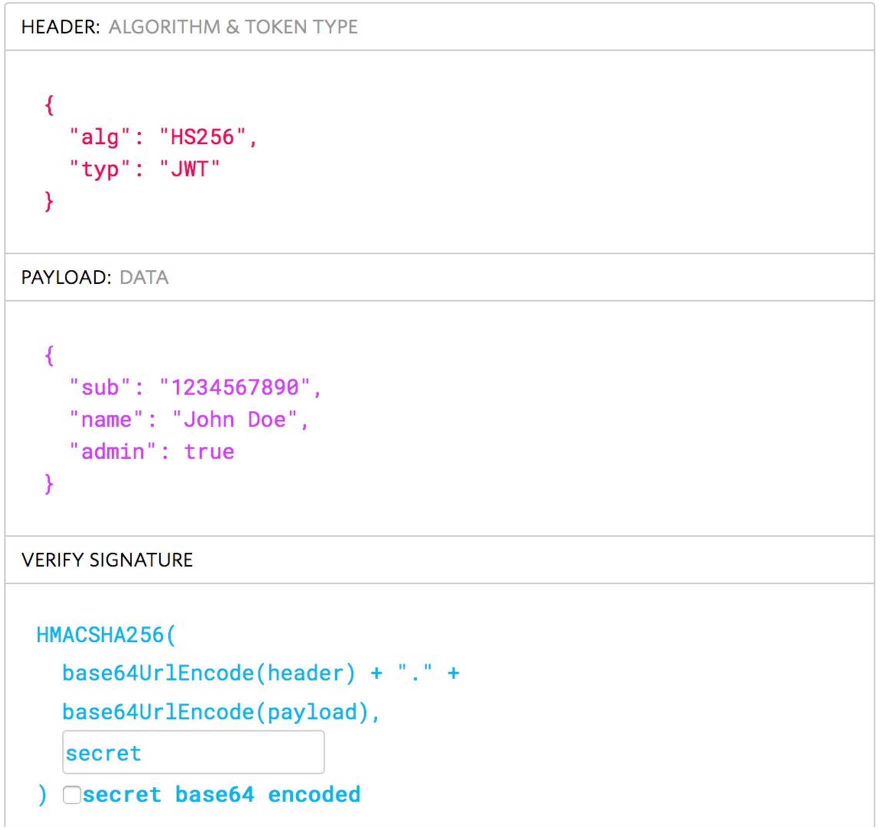


- ➔ Applications or relying parties need to talk with identity providers securely.
- ➔ Each authentication has unique messages associated.
- ➔ Those message need to be secure in transit.

moz://a

JSON Web Token

Decoded



HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

PAYLOAD: DATA

```
{
  "sub": "1234567890",
  "name": "John Doe",
  "admin": true
}
```

VERIFY SIGNATURE

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  secret
) ☐secret base64 encoded
```

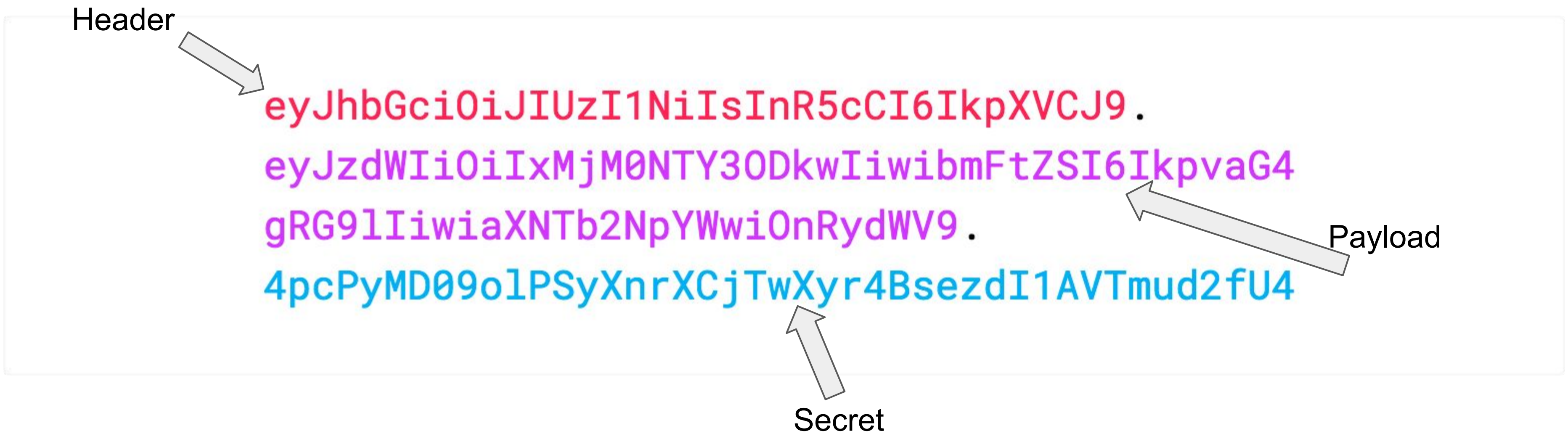
moz://a

JSON Web Token

Encoded

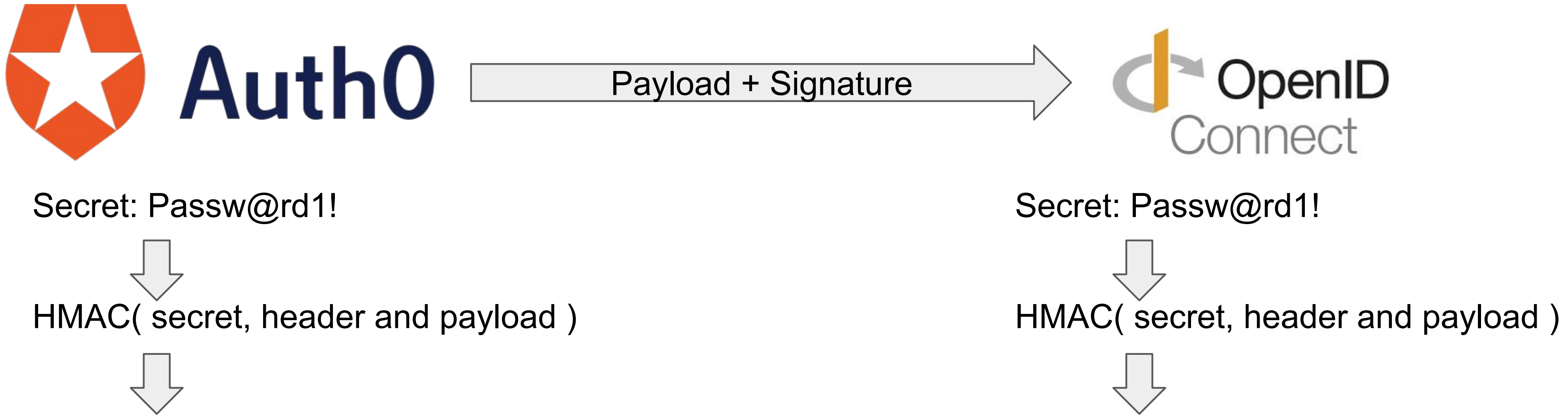
```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4
gRG9lIiwiaXNTb2NpYWwiOnRydWV9.
4pcPyMD09olPSyXnrXCjTwXyr4BsezdI1AVTmud2fU4
```

What’s a digital signature?



```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4
gRG9lIiwiaXNTb2NpYWwiOnRydWV9.
4pcPyMD09olPSyXnrXCjTwXyr4BsezdI1AVTmud2fU4
```

Digital Signatures Continued


Auth0
Payload + Signature
OpenID Connect
Secret: Passw@rd1!
HMAC(secret, header and payload)
Signature = 0xC0FF33C0FF33
Secret: Passw@rd1!
HMAC(secret, header and payload)
Signature = 0xC0FF33C0FF33

If you did catch all that…

Summary :

Secure because math...

moz://a

What would that look like in code?

```
#This is the payload we receive in python

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6Ik1pY2hh
ZWwgVmFuIEtsZWVrIiwiYWRtaW4iOnRydWV9.puDI94cptsSD3STETIqT4MO84nA54P2VtT_i
H-mcu7I
```

First we have to split it apart…

```
#!/usr/bin/python

payload =
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6Ik1pY2hhZWwgVmFuIEtsZWVrIiwiYWRtaW4iOnRydWV9.puDI94cptsSD3STETIqT4MO84nA54P2VtT_iH-mcu7I

header = payload.split(‘.’)[0]

payload = payload.split(‘.’)[1]

signature = payload.split(‘.)[2]
```

moz://a

Now we need to sign it…

```
#!/usr/bin/python
import hmac
import hashlib

payload = ( redacted for brevity )
secret = ByRzU2haBzT0dLwt7QZgzut4LqSPc5JW

header = payload.split(‘.’)[0]
payload = payload.split(‘.’)[1]
signature = payload.split(‘.)[2]

message = header + payload
digest_maker = hmac.new(secret, ‘ ’,hashlib.sha256)

this_signature = digest_maker.update(message).hexdigest()
```

Checking Signatures

```
#!/usr/bin/python
import hmac
import hashlib
 ** redacted for brevity **
message = header + payload
digest_maker = hmac.new(secret, ‘ ’,hashlib.sha256)

this_signature = digest_maker.update(message).hexdigest()

if this_signature == signature:
        #do things like trust the payload
else:
        #do things like access denied
```

A Defense in Depth Strategy

- ➔ TLS Certificates
- ➔ Application Security
- ➔ Custom Authorizers
- ➔ 2FA (Duo, OTP, etc.)

More Resources

JSON Web Tokens: https://jwt.io/introduction/

OpenIDC Security Best Practices:

https://wiki.mozilla.org/Security/Guidelines/OpenID_Connect

OAuth, OIDC, SAML, WS-Fed Comparison (blog by Niraj Bhatt)

Protect the Web- Get Involved With Mozilla!

Run Firefox Nightly!

https://www.mozilla.org/en-US/firefox/channel/desktop/

Contribute to the Mozilla Code Base!

https://developer.mozilla.org/en-US/docs/Mozilla/Developer_guide

Come work with us!

https://careers.mozilla.org/

Thanks!

Michael Van Kleeck

Twitter @michaelvkpdx

E-mail mvk@mozilla.com

moz://a

#mozilla

#DevRoadshow

on Periscope @mozilla

Q&A