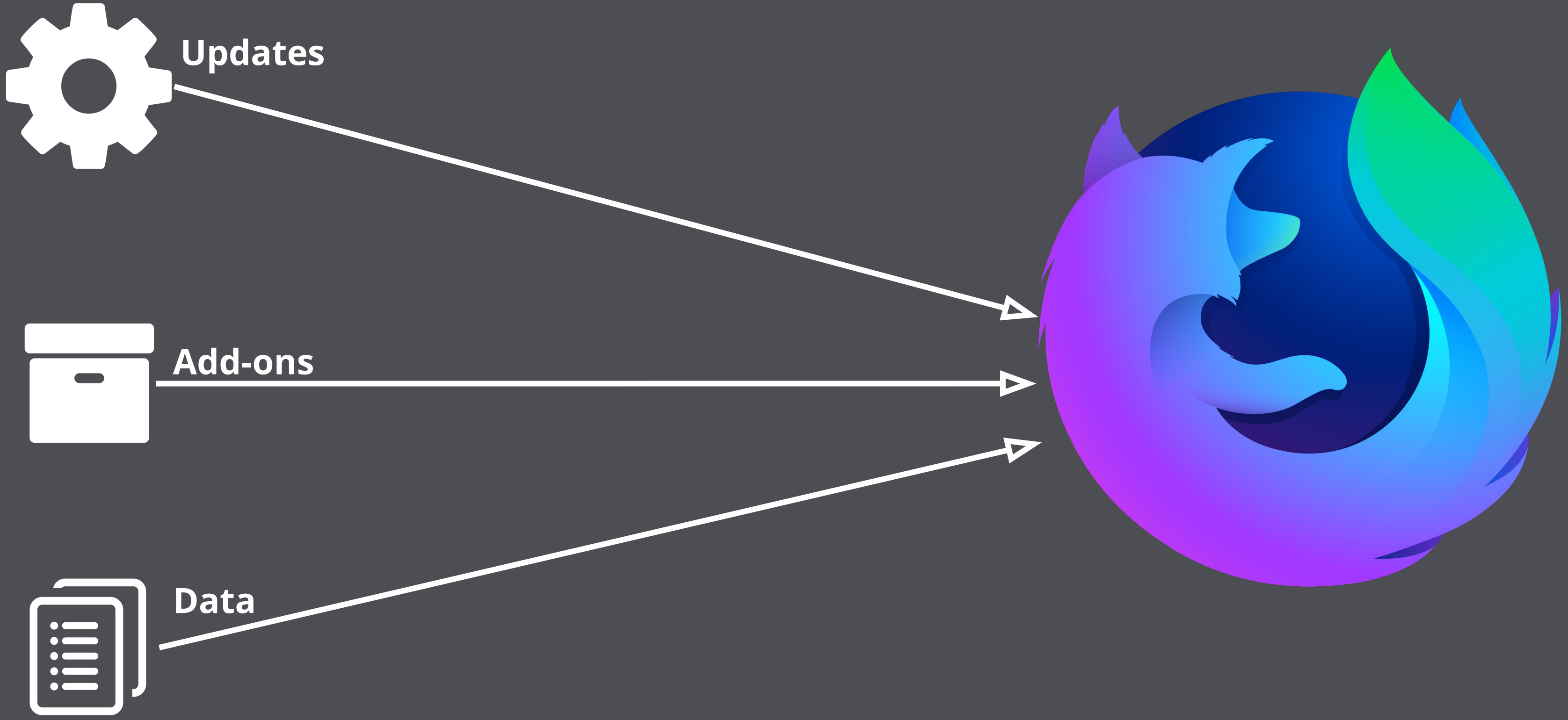
moz://a

Protecting Firefox data with Content Signature

Updating Firefox



Updates Security

- Updates are signed with PKCS1 using hard coded RSA keys

- Add-ons are signed with PKCS7 using an internal PKI

- Data signing? no good solution...

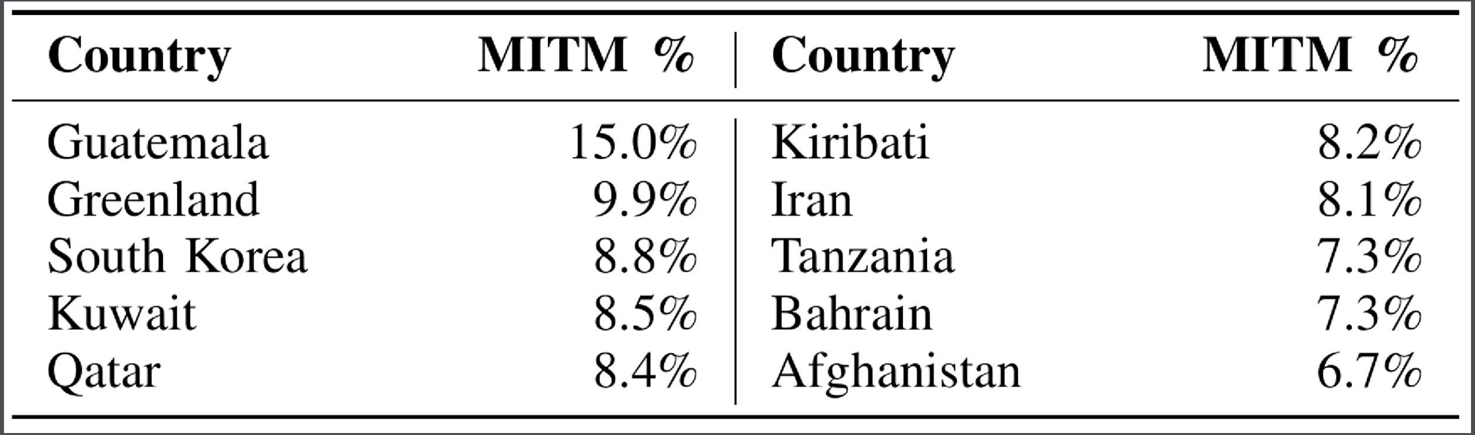
Serving data through web APIs

Industry best-practice: HTTPS and trust the backend. That has two problems:

1. HTTPS interception
2. Compromise of the web frontend

HTTPS Interception

- 4% of Firefox Updates are being intercepted
  (source: The Security Impact of HTTPS Interception)



Country	MITM %	Country	MITM %
Guatemala	15.0%	Kiribati	8.2%
Greenland	9.9%	Iran	8.1%
South Korea	8.8%	Tanzania	7.3%
Kuwait	8.5%	Bahrain	7.3%
Qatar	8.4%	Afghanistan	6.7%

Compromise of web API

- Written using modern web frameworks
  - Partially audited, change too often
  - Risks in the supply chain (insecure deps)

- Development agility vs security

- Better model: reduce security pressure by signing data in air-gapped backend

Content Signature

A Content Signature guarantees the integrity of data collections sent to Firefox

It does not

- protect confidentiality
- protect availability
- replay of prior revisions

Content Signature

A Content Signature is

- an ECDSA P-384 signature
- on the SHA2-384 hash of the data
- encoded using DL/ECSSA representation of the R and S values
- in Base64 URL Safe

```
9_YUTeoubIAcWX5TzjB2INOV1_E9KZfIrJsa6uFqTlL_XmPb2lj_qY2n3BRJZ1sfZ
Hf033JqO14yKEiv3iwzuveWQjSGqfYnSAzW7PiCrJXMfHXoVVEsLknzhyAcRww1
```

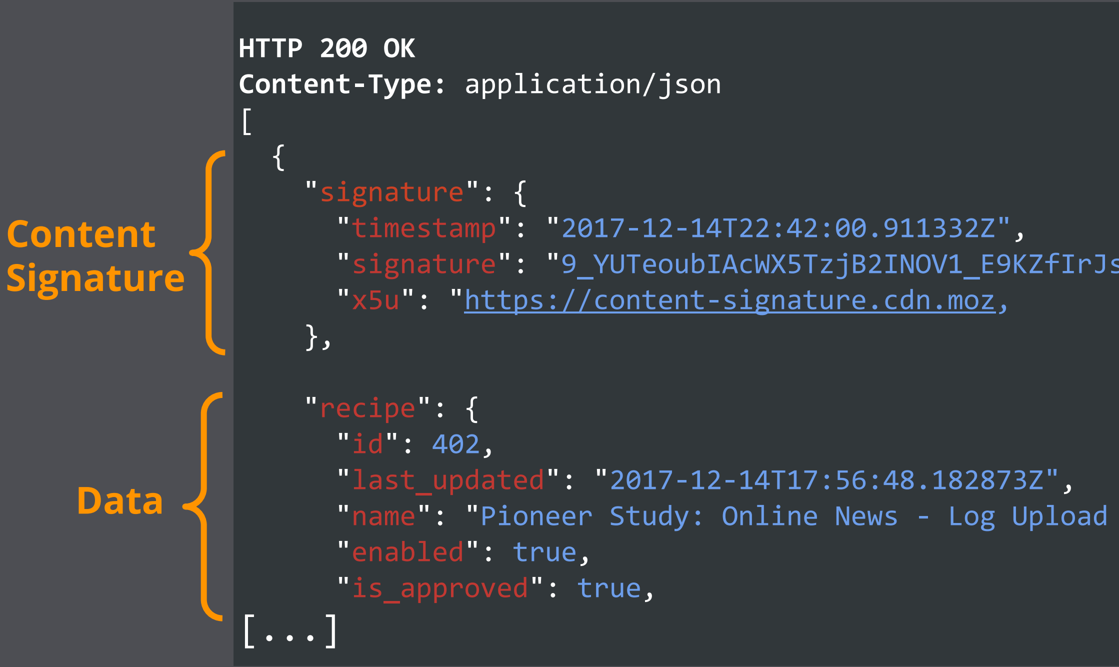
Internal Firefox PKI

- End-entity signing certs are issued by an internal PKI, same as add-ons
- Intermediate certs are constrained to *.content-signature.mozilla.org
- Firefox downloads the cert chain using an x5u value in the signature (hash of the root is hardcoded).

Delivering Content Signatures

Two methods:

- HTTP response header
- Signature field in API response

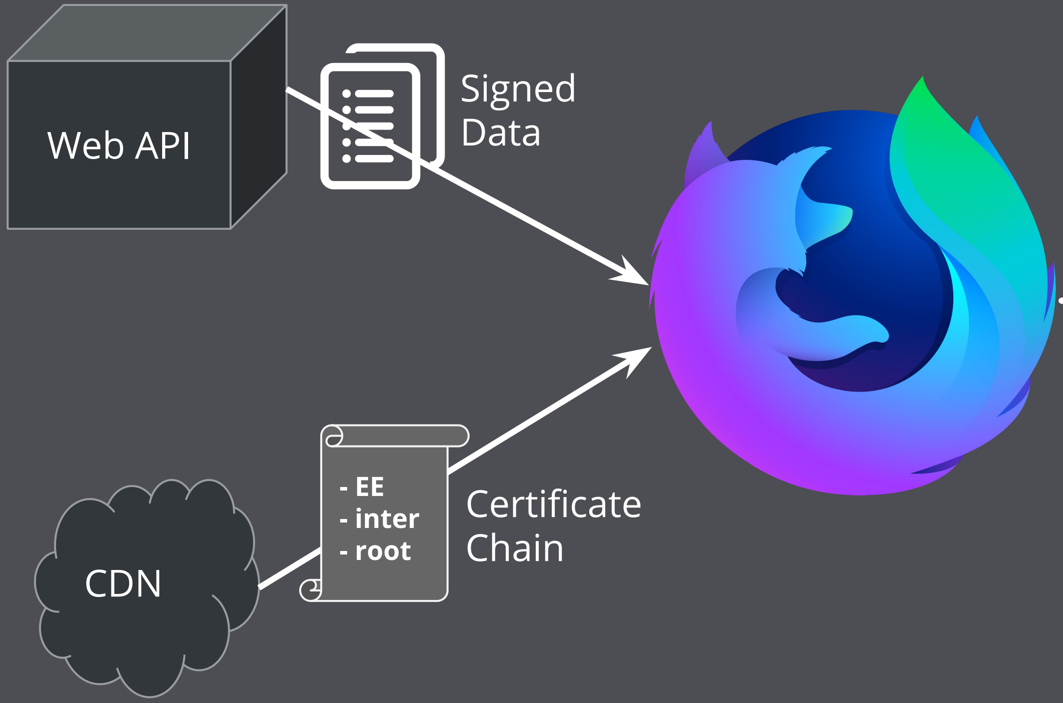


```
HTTP 200 OK
Content-Type: application/json
[
  {
    "signature": {
      "timestamp": "2017-12-14T22:42:00.911332Z",
      "signature": "9_YUTeoubIAcWX5TzjB2INOV1_E9KZfIrJ
      "x5u": "https://content-signature.cdn.moz,
    },

    "recipe": {
      "id": 402,
      "last_updated": "2017-12-14T17:56:48.182873Z",
      "name": "Pioneer Study: Online News - Log Upload
      "enabled": true,
      "is_approved": true,
[...]
```

moz://a

Verifying Content Signatures



- Parse Signature
- Retrieve chain
- Parse chain
- Verify chain/root
- Compare EE CN with app name
- Verify ECDSA sig

Operational Security

- Only air-gapped backends can talk to the signing service, no public access
- Signing certs are moderately short lived (90 days) to reduce risk of a leaked old cert being reused fraudulently
- PKI root is stored in offline HSMs

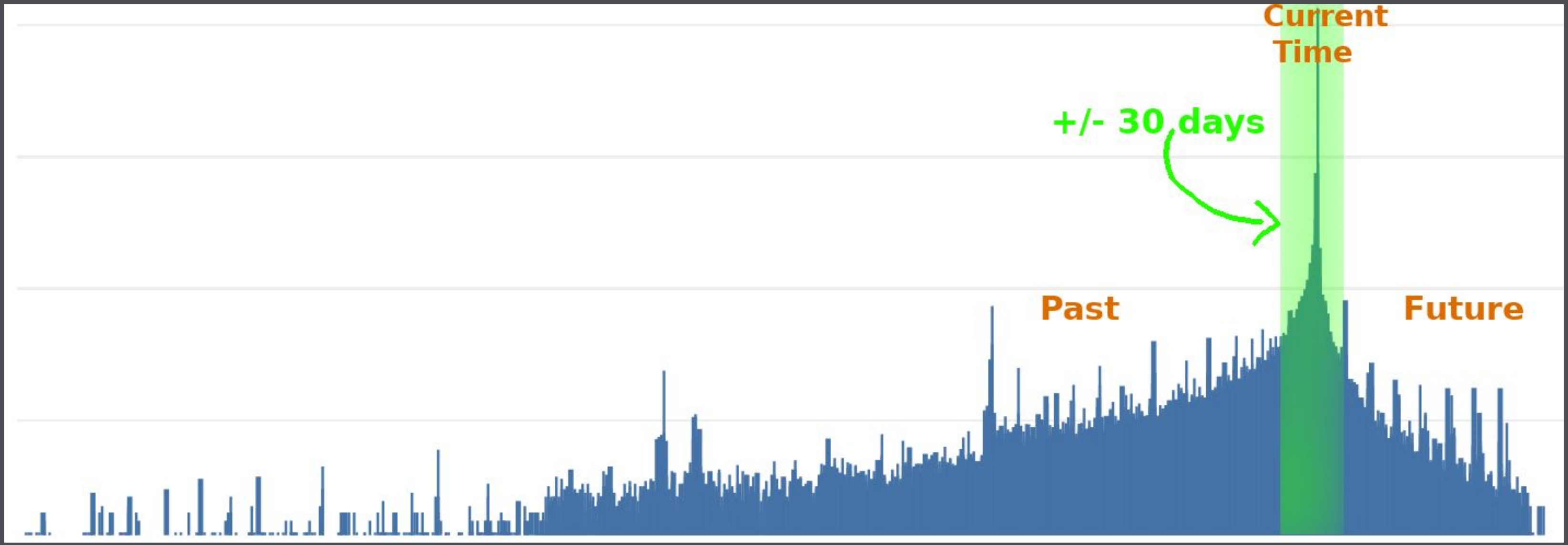
moz://a

Some interesting problems

- Checking certificate validity
- Measuring verification failures
- Preparing for emergency revocations

Checking certificate validity

- 1.2% of clients have bad clocks, most within 30 days
- 0.11% have clocks beyond 30 days



Logarithmic scale of Firefox clients with sorted by clock accuracy, in days

Checking certificate validity

- Signature verification fails when client clock is outside of cert validity

- We enforce validity checks, meaning clients with bad clocks don’t get the data

- Limit impact by using 90 days certs with 30 days wiggle room before & after

Measuring validation failures

- Firefox drops the data when the signature does not validate

- Getting a ping when that happens is critical to debugging
  - We plan to use Firefox Telemetry to get a ping when a signature fails
  - Future work: identify how/why that happens

moz://a

Emergency revocations

- Revoking a leaked end-entity or intermediate can use OneCRL
    - takes a few minutes and propagates quickly
    - Side note: OneCRL is also signed using Content Signature
- Revoking the root takes a Firefox update, which still uses separate hardcoded keys.

Implementation complexity

- Moderate initial effort, ongoing maintenance is lightweight
- +800 LOC in Firefox; 4000 LOC in Backend
- Fairly small team
    - Julien Vehent
    - Franziskus Kiefer
    - Bob Micheletto
    - Mark Goodwin
    - Martin Thomson
    - Remy Hubscher
    - Michael Cooper
    - Nan Jiang

Thank You!

Check out the code, it's online:

- Backend at **go.mozilla.org/autograph**
- Firefox verification code is under **security/ manager/ ssl/ ContentSignatureVerifier.cpp**