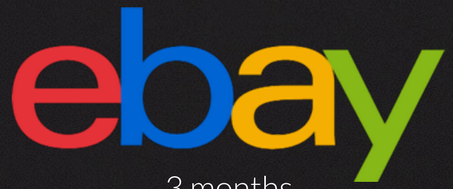
NETFLIX

PKI at Scale Using Short-Lived Certificates

Bryan D. Payne Engineering Manager, Platform Security

*Notified via extortion attempt

1 month



3 months

13 months

17 months

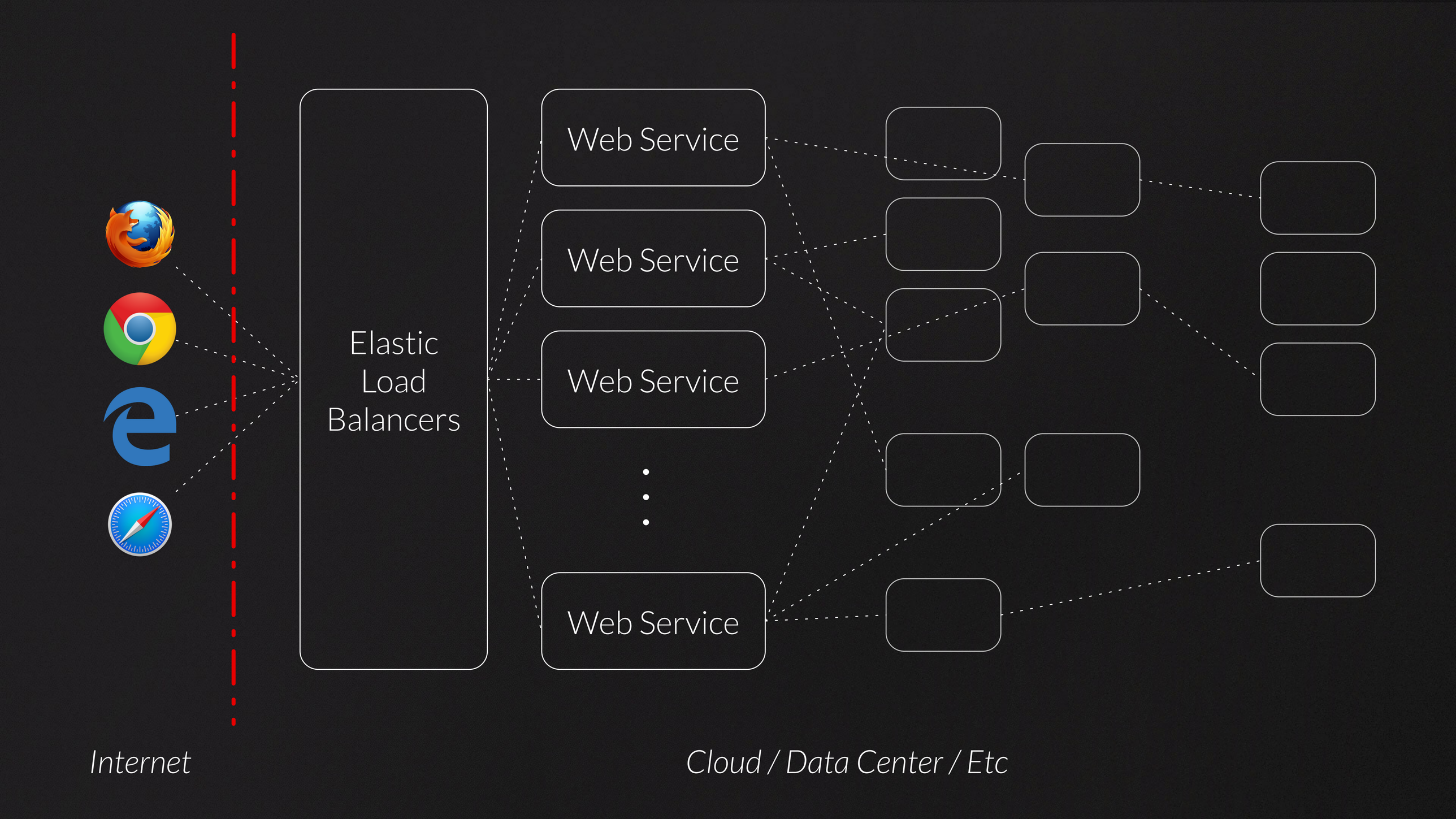
2 weeks

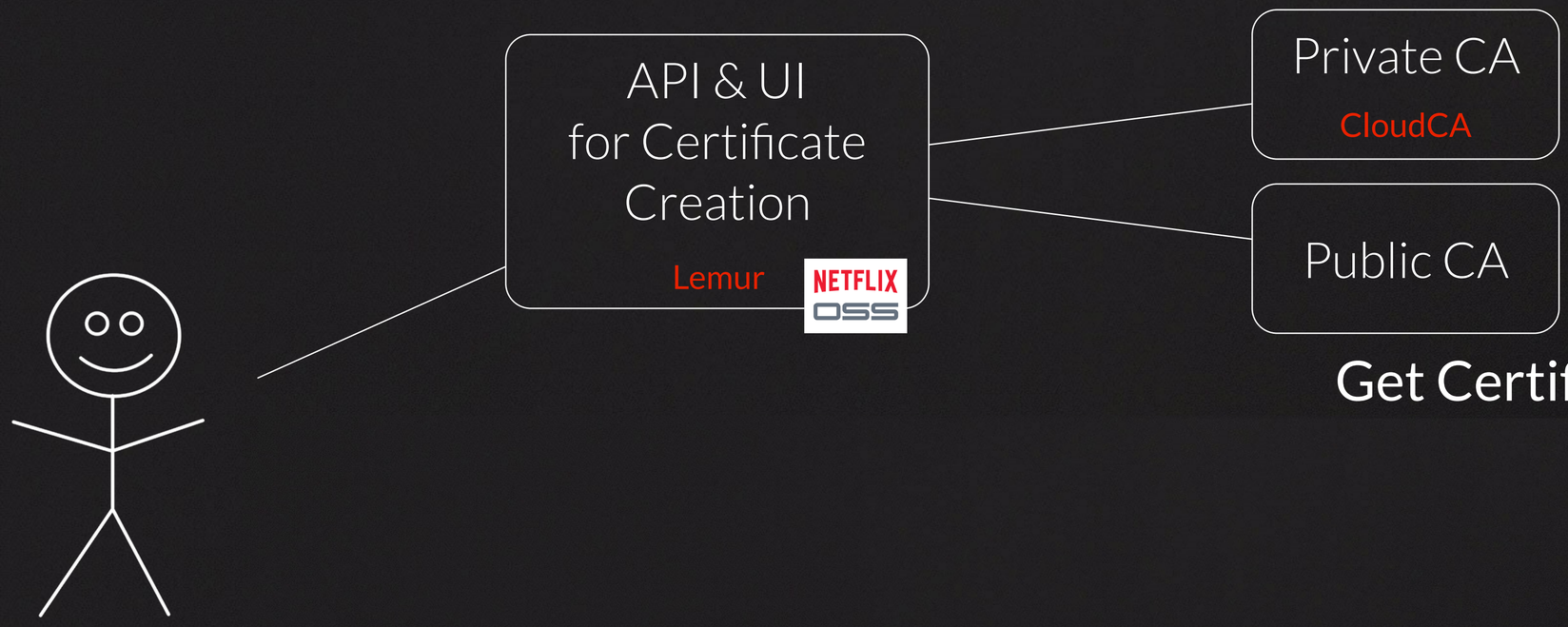
4 months

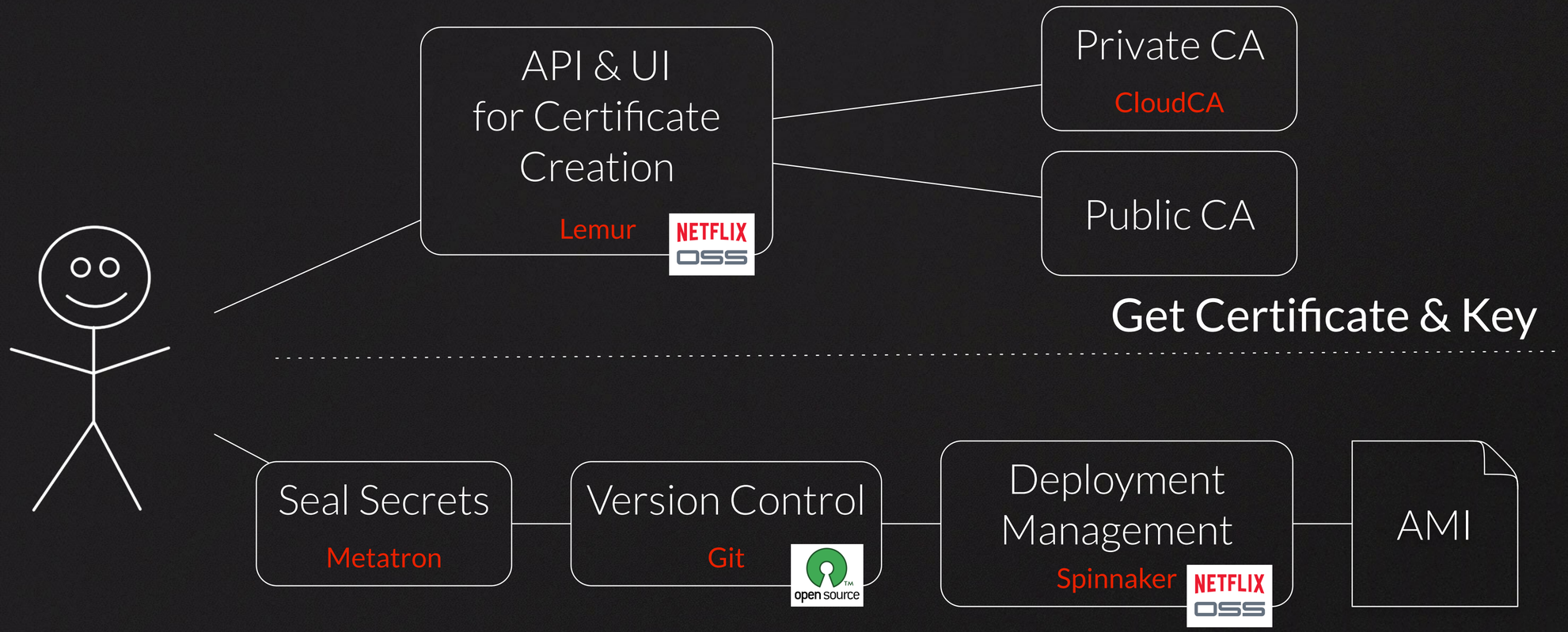
3 weeks

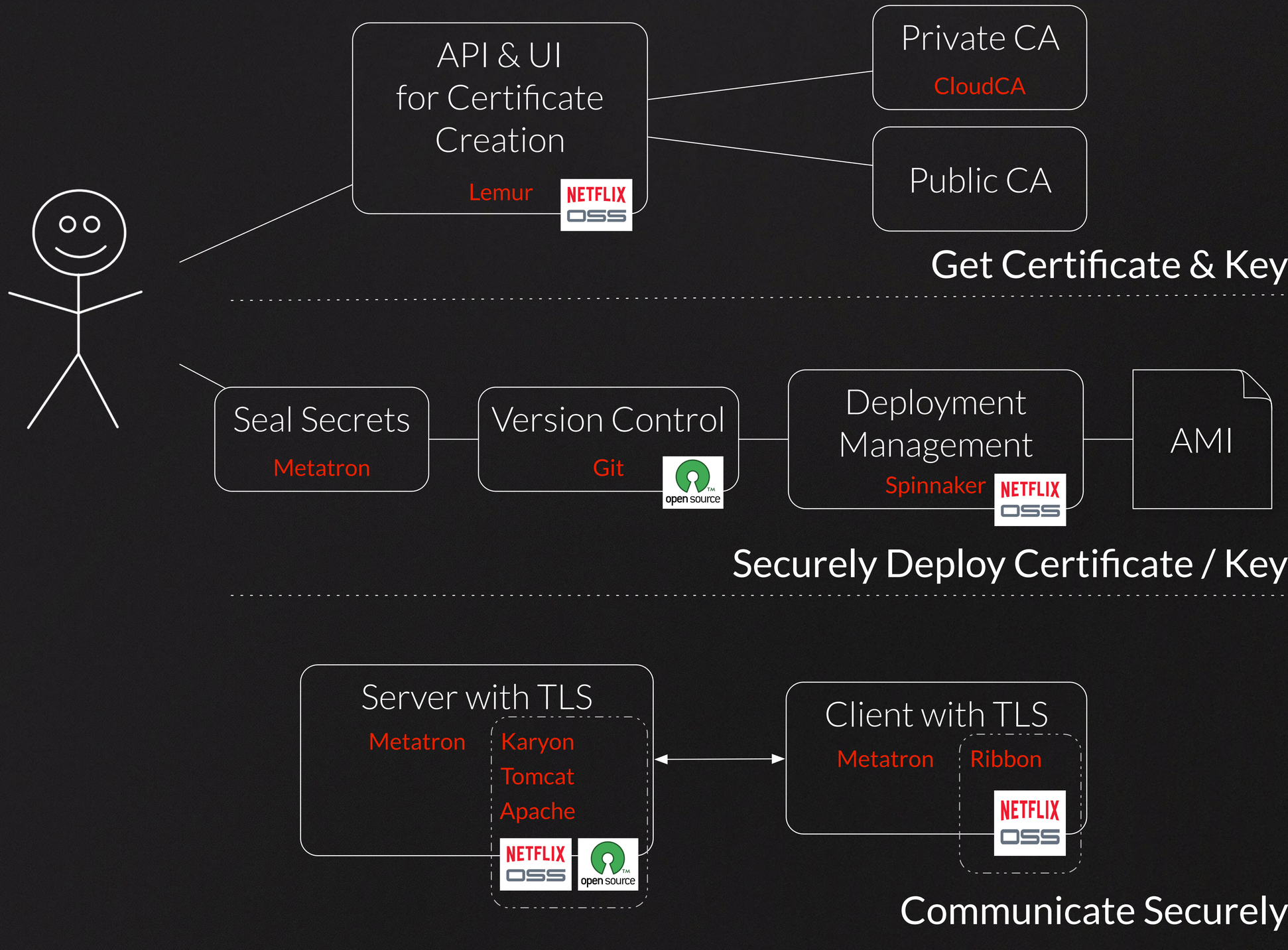
8 months

weeks | <6 months | 6-12 months | >1 year


Elastic
Load
Balancers
Web Service
Web Service
Web Service
Web Service
Internet
Cloud / Data Center / Etc


API & UI
for Certificate
Creation
Lemur
NETFLIX
OSS
Private CA
CloudCA
Public CA
Get Certificate & Key


API & UI
for Certificate
Creation
Lemur
NETFLIX
OSS
Private CA
CloudCA
Public CA
Get Certificate & Key
Seal Secrets
Metatron
Version Control
Git
open source
Deployment
Management
Spinnaker
NETFLIX
OSS
AMI
Securely Deploy Certificate / Key


API & UI
for Certificate
Creation
Lemur
NETFLIX
OSS
Private CA
CloudCA
Public CA
Get Certificate & Key
Seal Secrets
Metatron
Version Control
Git
open source
Deployment
Management
Spinnaker
NETFLIX
OSS
AMI
Securely Deploy Certificate / Key
Server with TLS
Metatron
Karyon
Tomcat
Apache
NETFLIX
OSS
open source
Client with TLS
Metatron
Ribbon
NETFLIX
OSS
Communicate Securely

Revocation Is Hard

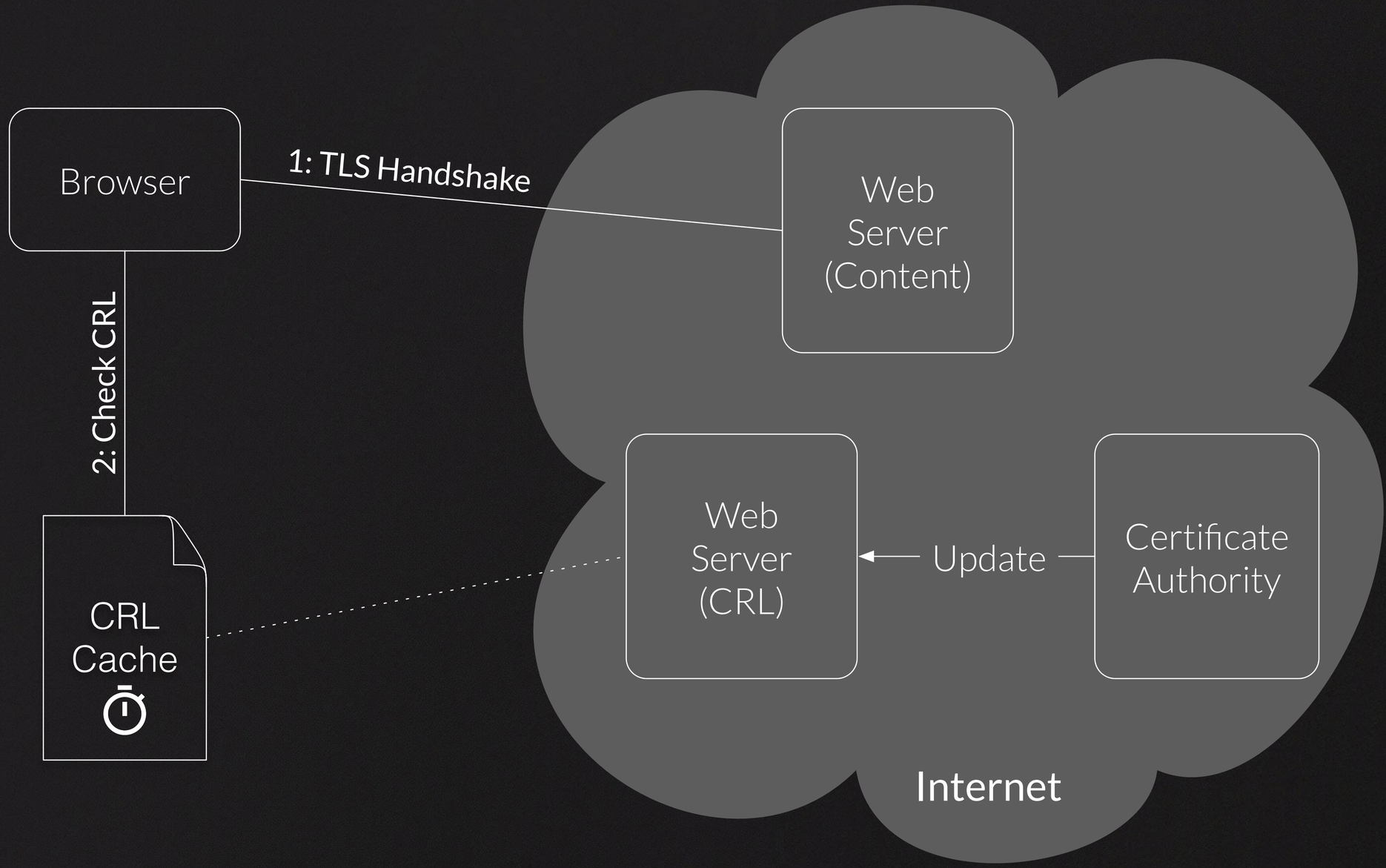
CRL (rfc2459)

OCSP (rfc2560)

OCSP stapling (rfc6066)

OCSP must staple (draft-hallambaker-muststaple-00)

CRL: Certificate Revocation List



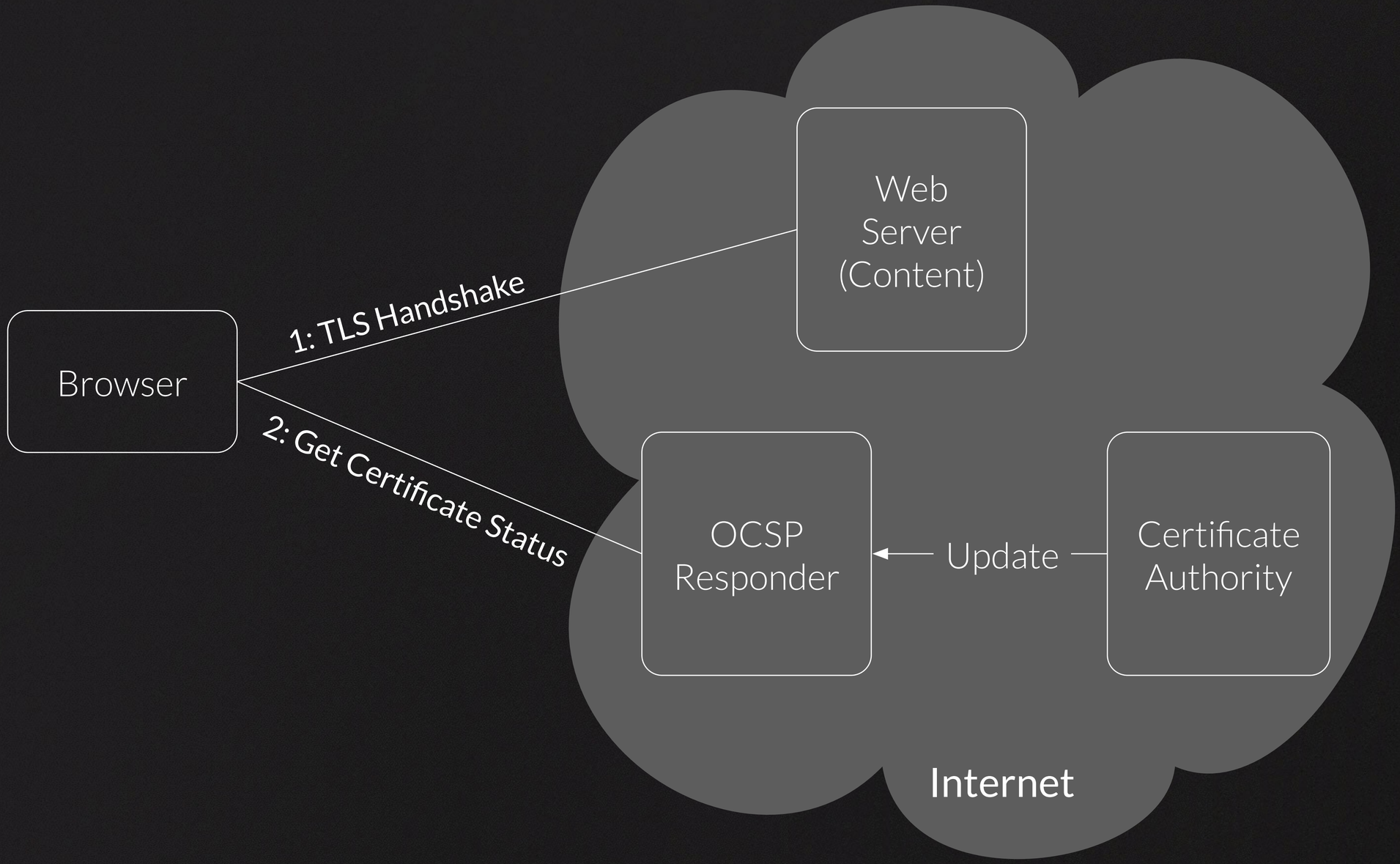
~~CRL (rfc2459)~~

OCSP (rfc2560)

OCSP stapling (rfc6066)

OCSP must staple (draft-hallambaker-muststaple-00)

OCSP: Online Certificate Status Protocol



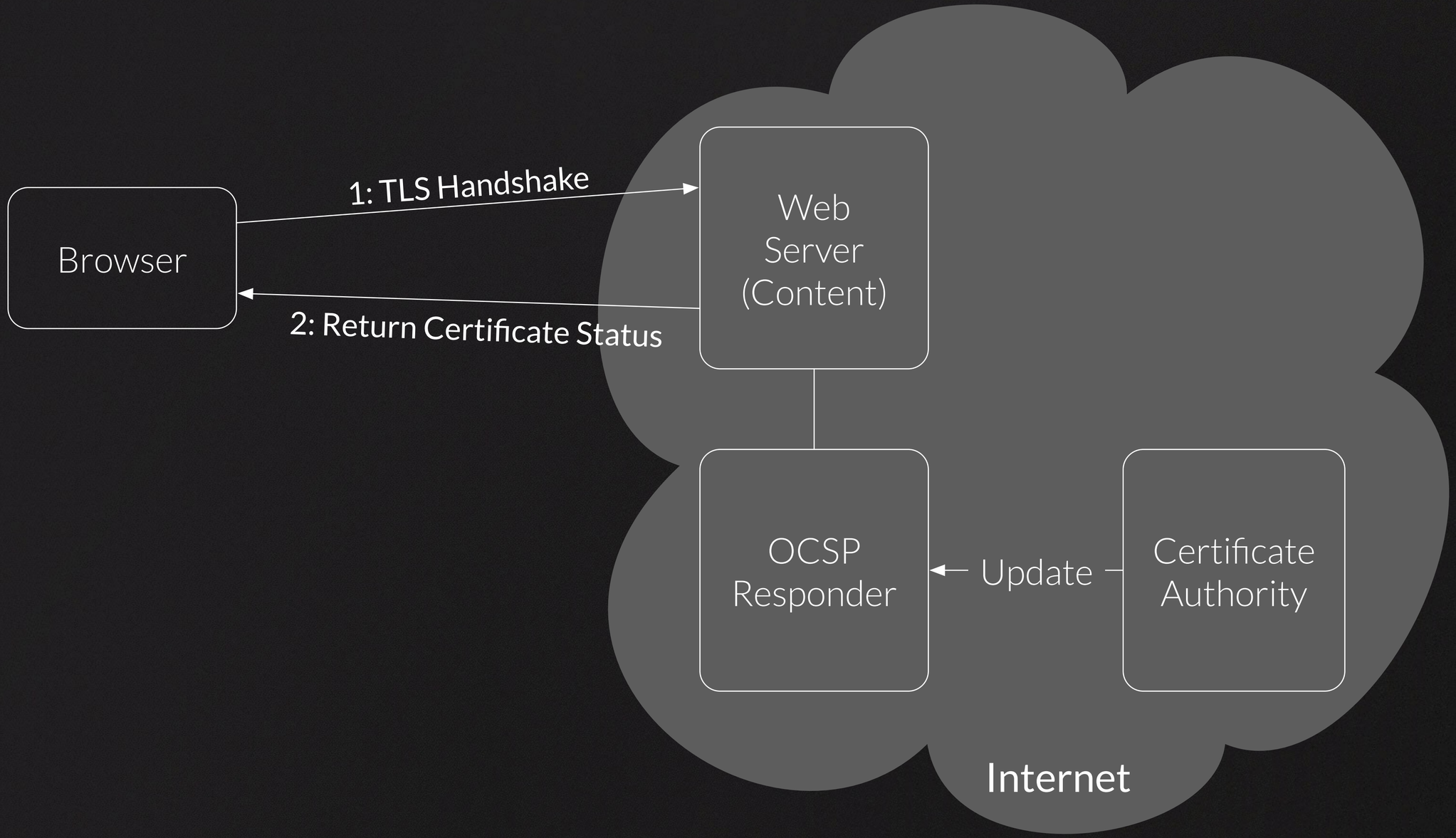
~~CRL (rfc2459)~~

~~OCSP (rfc2560)~~

OCSP stapling (rfc6066)

OCSP must staple (draft-hallambaker-muststaple-00)

OCSP Stapling



~~CRL (rfc2459)~~

~~OCSP (rfc2560)~~

~~OCSP stapling (rfc6066)~~

OCSP must staple (draft-hallambaker-muststaple-00)

	OCSP must-staple	OCSP staple	OCSP	CRL
Java				
C				
Python				
JavaScript				

M Georgiev et al., “The most dangerous code in the world: validating SSL certificates in non-browser software”, In Proceedings of ACM CCS, 2012.


IN GOD WE TRUST
LIBERTY
2013
S

Photo Credit: Kayamon (CC BY-SA 3.0)
https://en.wikipedia.org/wiki/File:Penny_Harvest_Field_2007.jpg

6 MONTHS

4 DAYS

1 MONTH

Short-Lived Certificates

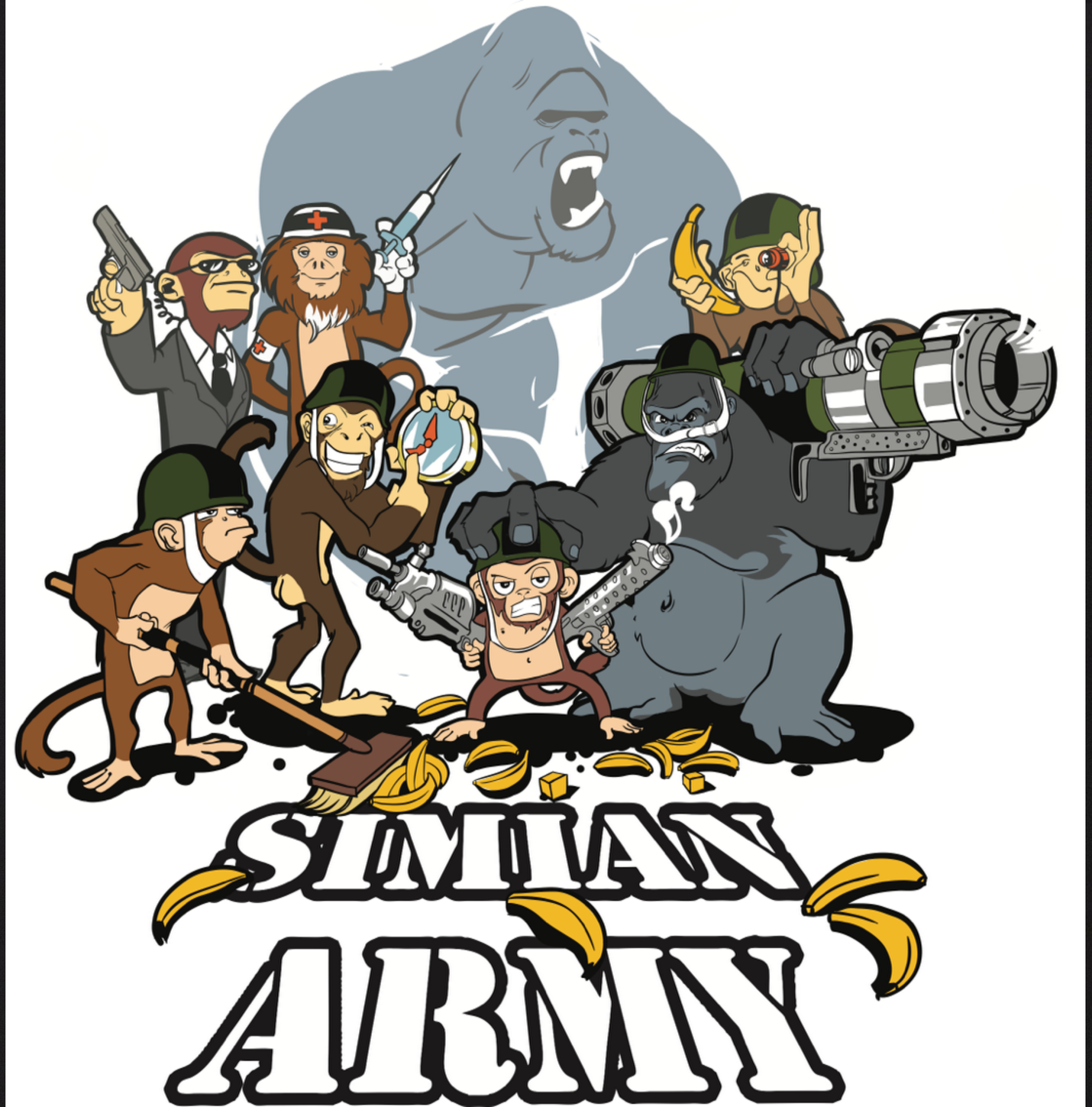
4 HOURS

3 MONTHS

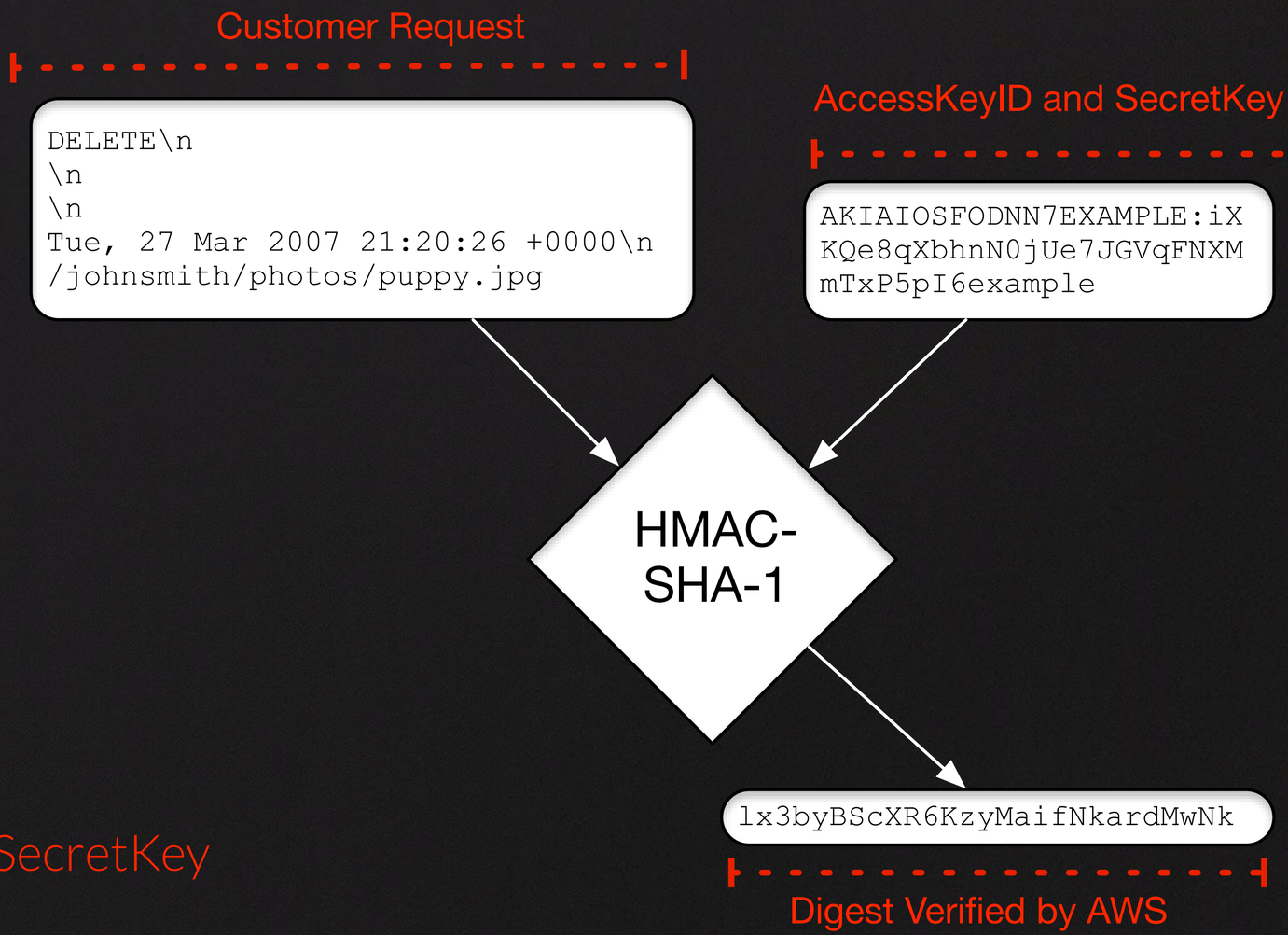
1 WEEK

- R Rivest, “Can We Eliminate Certificate Revocation Lists?”, In *Proceedings of Financial Cryptography*, 1998.
- E Topalovic et al., “Towards Short-Lived Certificates”, In *Proceedings of IEEE Oakland Web 2.0 Security and Privacy* (W2SP), 2012.

Photo Credit: Bhernandez (CC BY 2.0)
https://www.flickr.com/photos/kennyuhh/2917293212


SIMIAN
ARMY

AWS HMAC Generation



Lifecycle of AccessKeyID and SecretKey is of utmost interest here.

Not real secret keys, sorry.

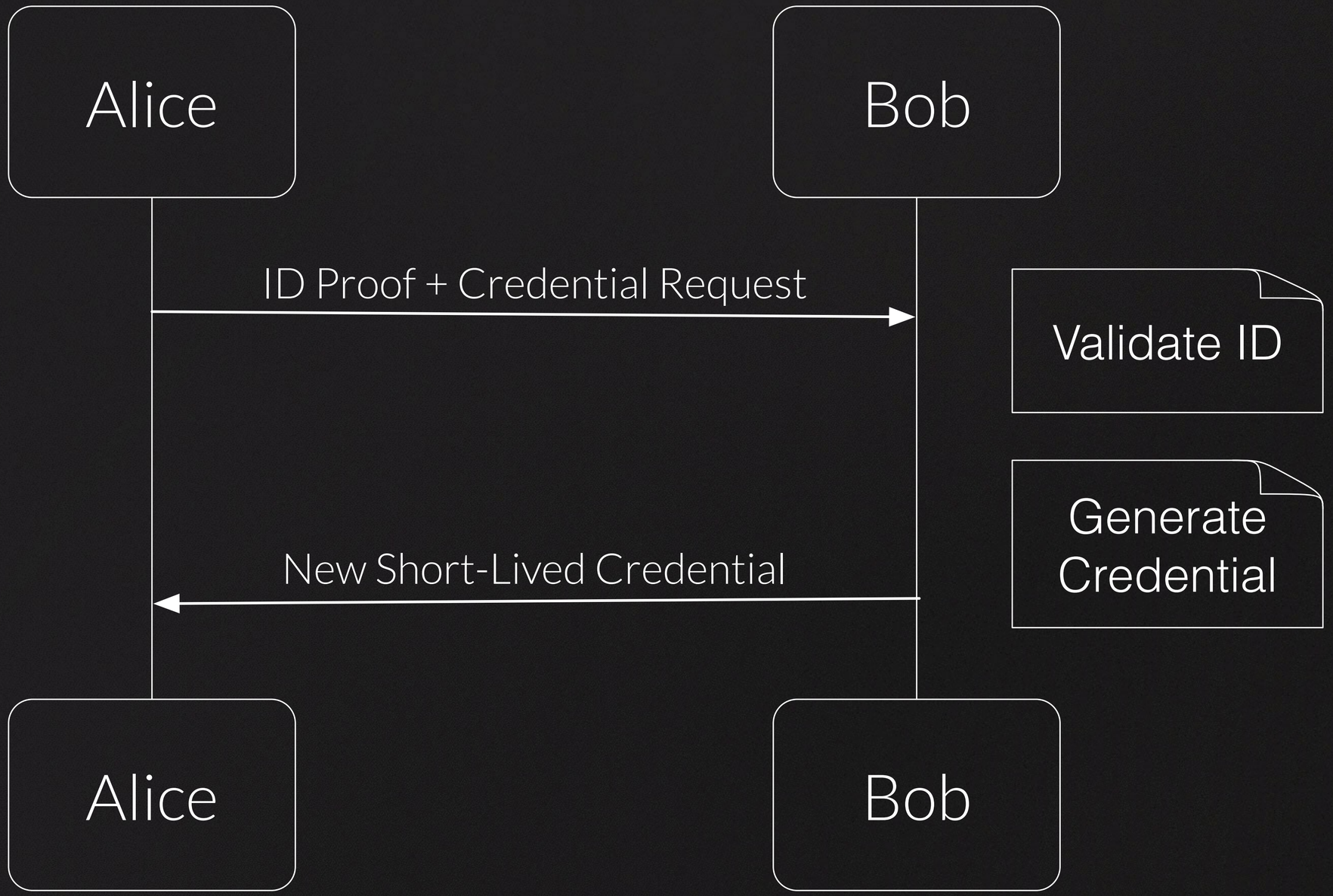
Circa 2012: AWS SDKs Introduce the Provider Paradigm

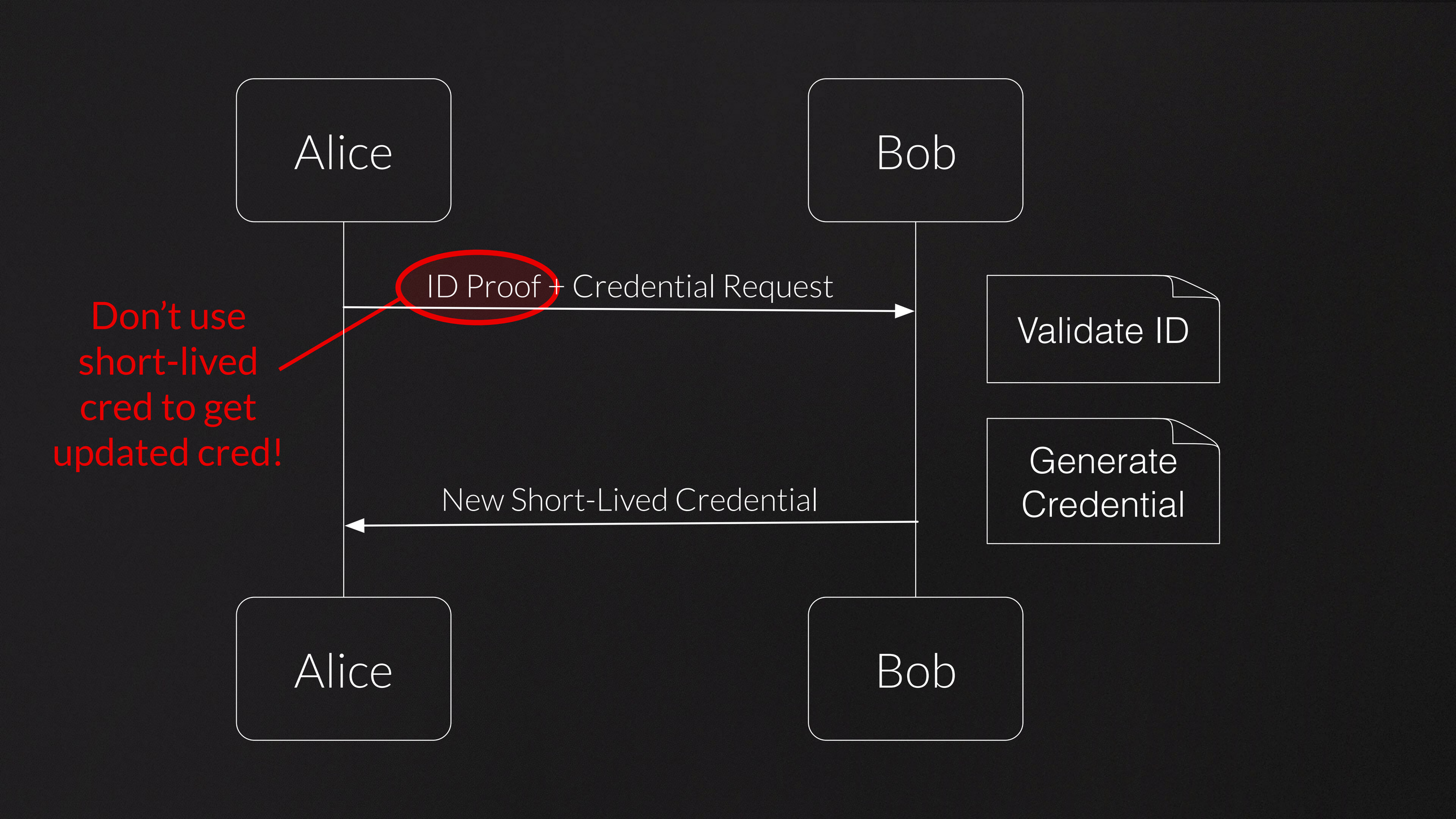
```
// provider paradigm dynamically asks for keys every time
AWSCredentialsProvider prov = new AWSCredentialsProvider(){
  public AWSCredentials getCredentials(){
      RESTfulObj AWSKey = RESTService.get(“server/getAWSKey”);
                return new BasicAWSCredentials(
                          AWSKey.getAccessID(), AWSKey.getSecretKey());
  }
};

AmazonSimpleDBClient client = new AmazonSimpleDBClient(prov);
client.listDomains();
```

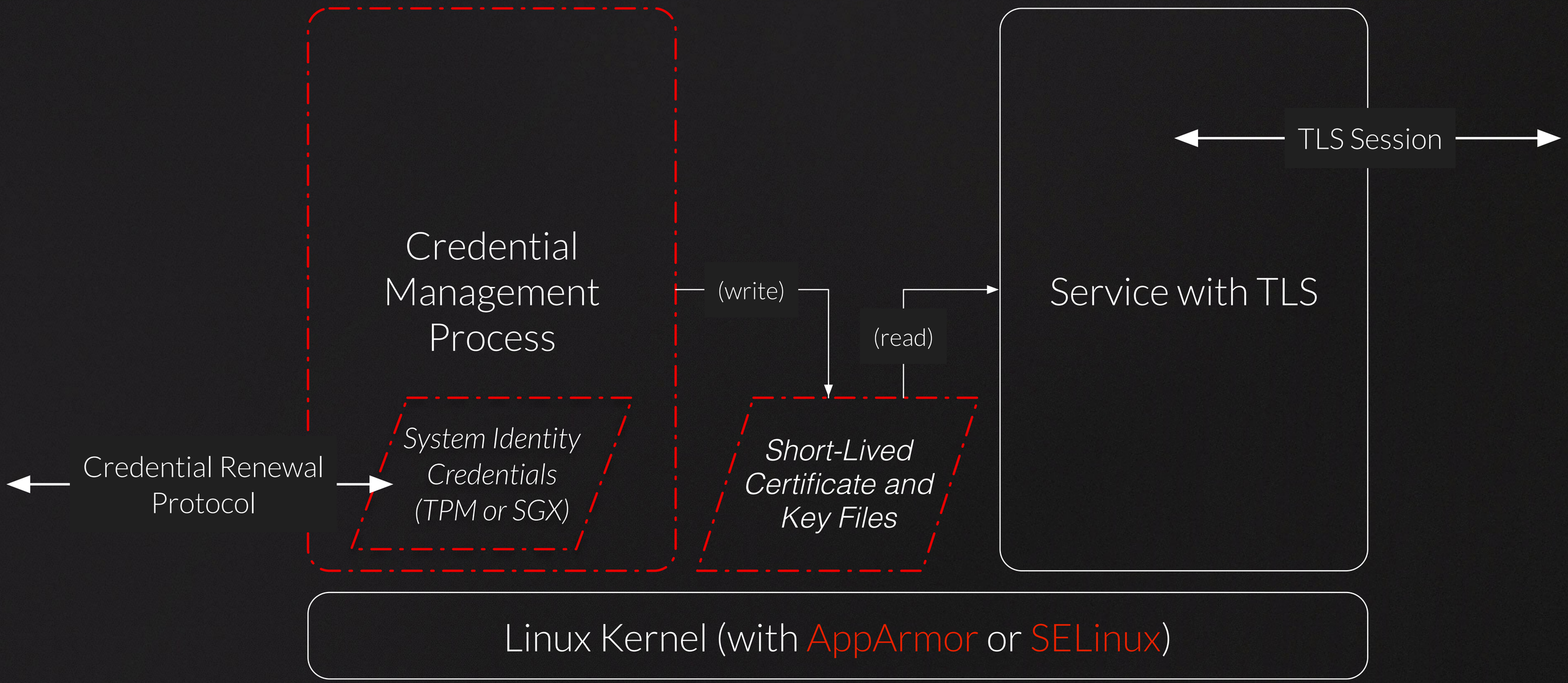
The client object in the above code example no longer caches keys.

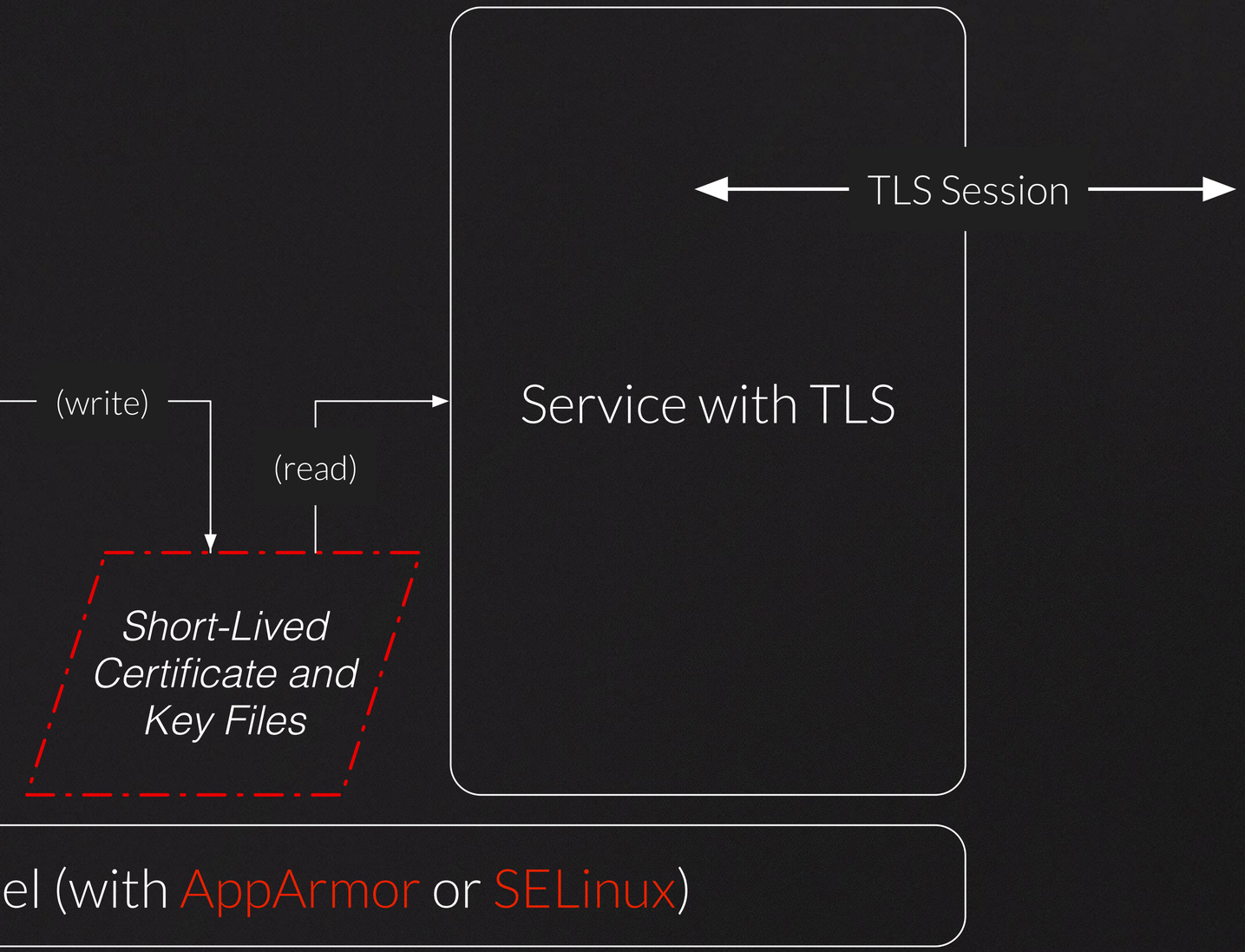
On Instance Credentials

```
$curl http://169.254.169.254/latest/meta-data/iam/security-credentials/role
{
  "Code" : "Success",
  "LastUpdated" : "2015-09-17T01:29:49Z",
  "Type" : "AWS-HMAC",
  "AccessKeyId" : "ASIAIL6IJJCXLEXAMPLE",
  "SecretAccessKey" : "iXKQe8qXbhnN0jUe7JGVqFNXMmTxP5pI6example",
  "Token" : "...",
  "Expiration" : "2015-09-17T07:47:45Z"
}
```


Alice
Bob
ID Proof + Credential Request
Validate ID
Generate Credential
New Short-Lived Credential
Alice
Bob


Alice
Bob
ID Proof + Credential Request
Don’t use short-lived cred to get updated cred!
Validate ID
Generate Credential
New Short-Lived Credential
Alice
Bob

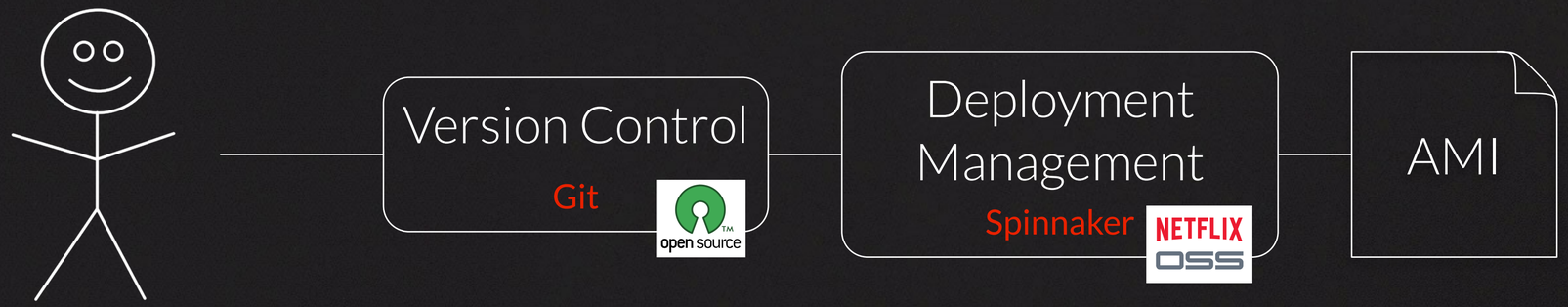

Credential Management Process
(write)
(read)
Service with TLS
TLS Session
Credential Renewal Protocol
System Identity Credentials (TPM or SGX)
Short-Lived Certificate and Key Files
Linux Kernel (with AppArmor or SELinux)



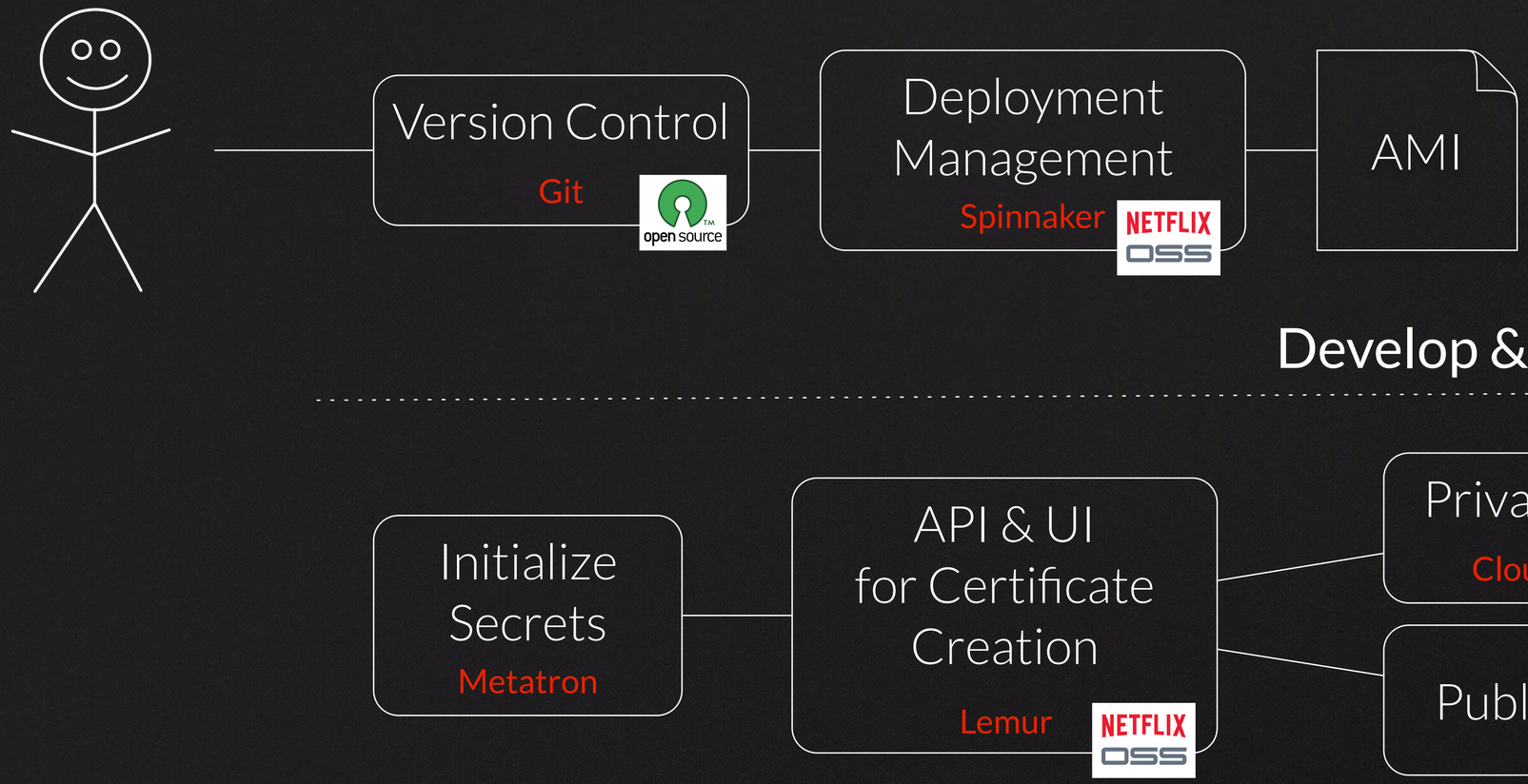
Loading new certificates into service…

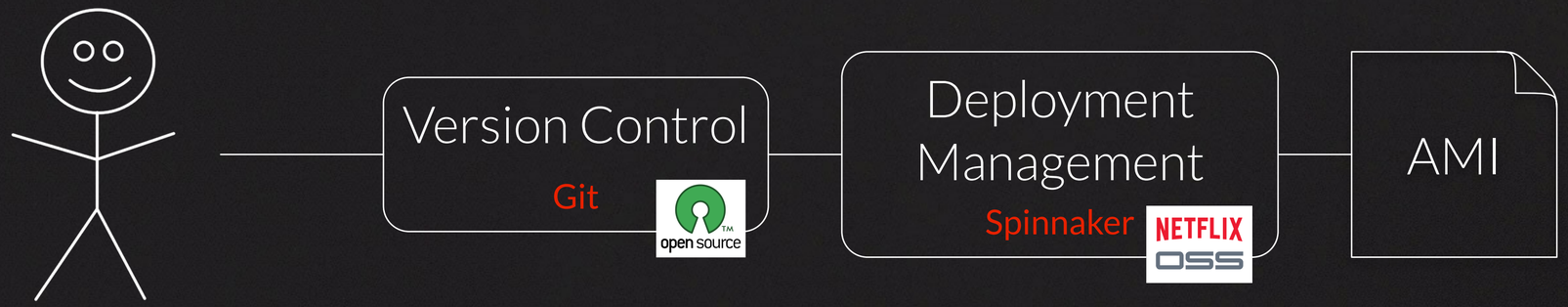
- Send signal to service
- Restart service
- Design service to reload certificates periodically

	How to load a new certificate and private key?	Zero downtime?
Apache	graceful restart	Maybe
Nginx	reload	Yes
Tomcat	restart	No
HAProxy	reload	No
Stunnel	HUP	No
Ghostunnel	SIGUSR1	Yes



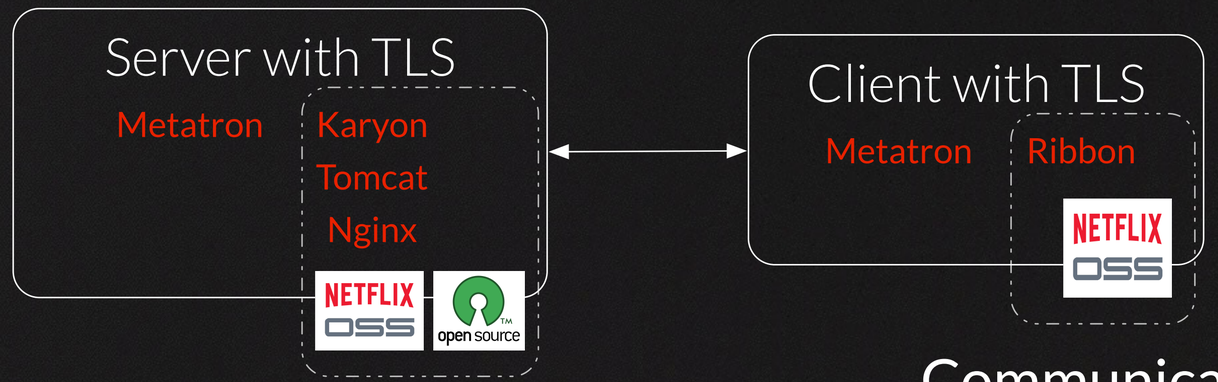
Develop & Deploy Code


Version Control
Git
open source
Deployment Management
Spinnaker
NETFLIX OSS
AMI
Develop & Deploy Code
Initialize Secrets
Metatron
API & UI for Certificate Creation
Lemur
NETFLIX OSS
Private CA
CloudCA
Public CA
Provision Credentials at Startup

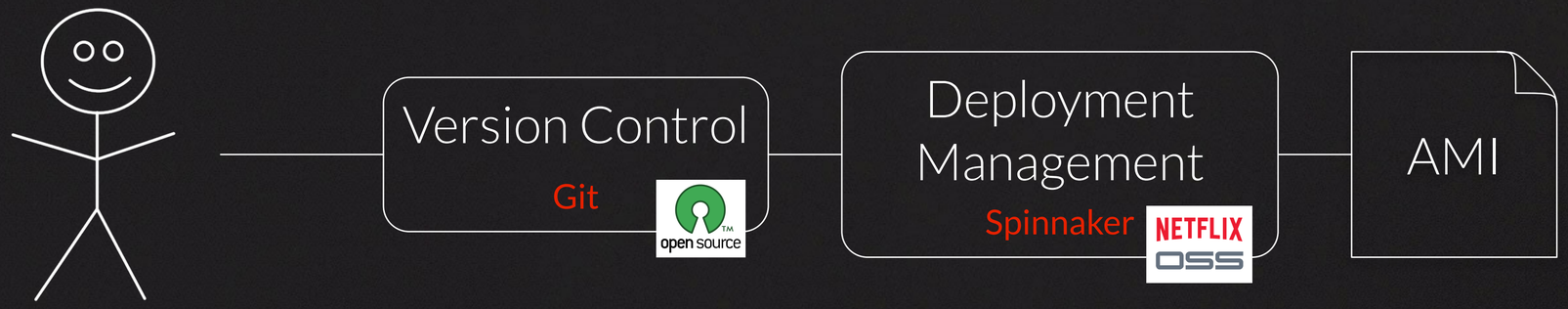


Develop & Deploy Code

Provision Credentials at Startup

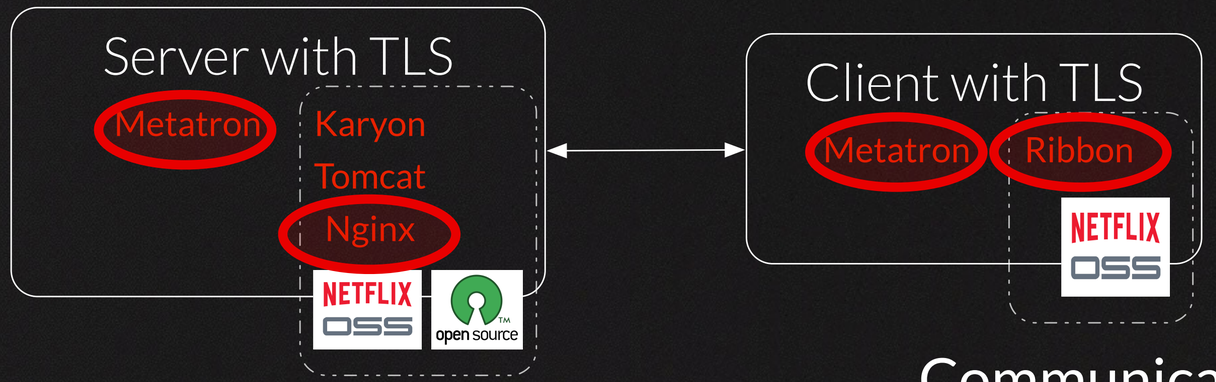


Communicate Securely



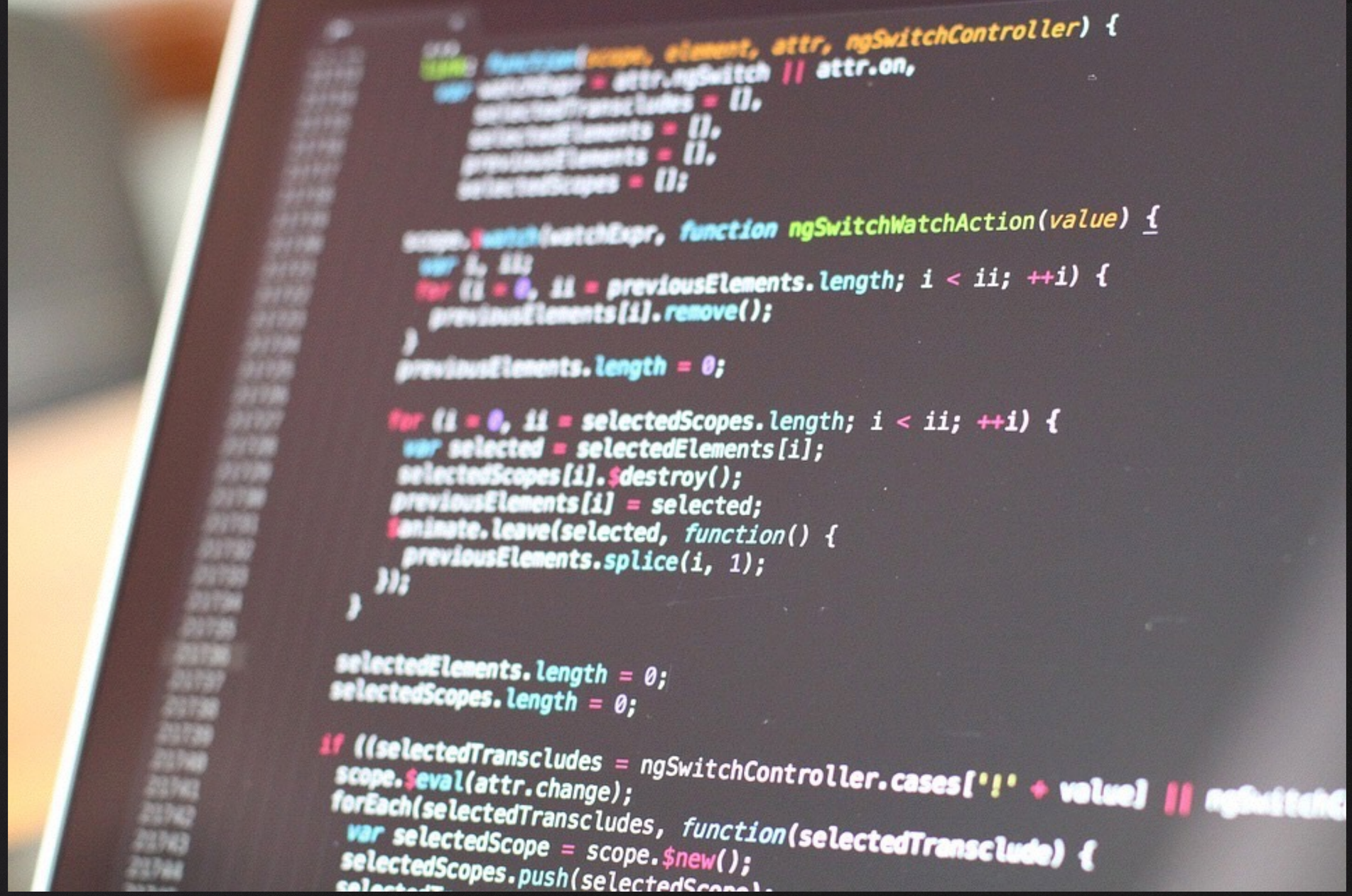
Develop & Deploy Code

Provision Credentials at Startup



Communicate Securely

Long-Lived Certificates	Short-Lived Certificates
• Improve attack detection, in practice • Retrofit your applications to support revocation	• Refresh certificates • Update server / client to support graceful reloading of certificates



From Vision to Reality…

Questions?

bryanp@netflix.com
http://bryanpayne.org

[PS… I’m hiring!]