Encrypt ALL the things with **LetsEncrypt**

Created by:
- ➔ Justin W. Flory
- ➔ Solomon Rubin

Introduction

What is TLS and why do I need it?

- TLS stands for Transport Layer Security
  - Difference between https and http
  - Encrypts communications with web servers on the fly
- Normally, purchase TLS certificate from **Certificate Authority**

Old problems with getting certificates

- Basic encryption is expensive (especially with multiple subdomains)
- Most certificate authorities (CAs) focus on **identity** or **organization verification**
    - Most sites only need **domain verification**

What is LetsEncrypt?!

- *Imagine a world* where encryption is everywhere and your online communications are always secure
  - LetsEncrypt offers solution to increase security of the web
- Free certificates
  - Providing *only* domain verification
    - At zero cost
  - Creates a safer Internet

Key Principles

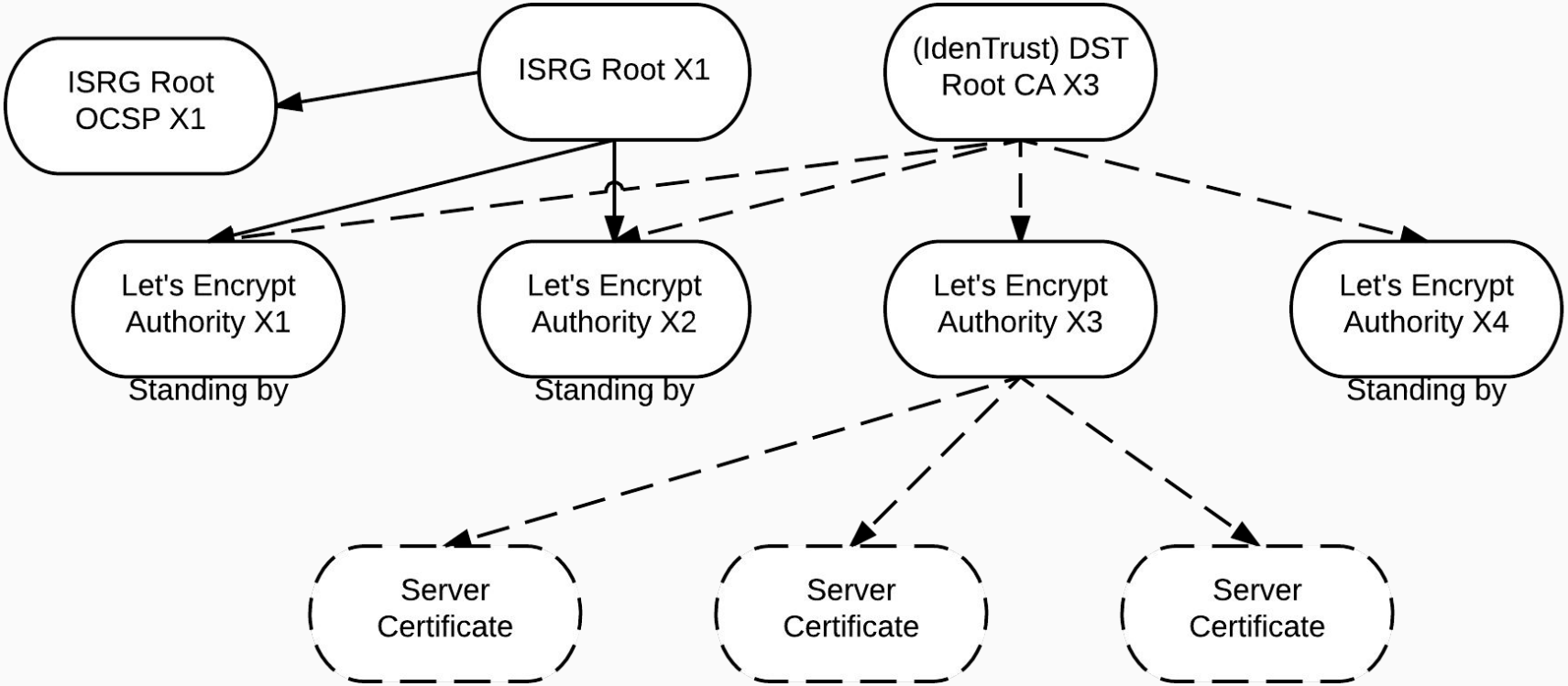
- **Free** for anyone who owns a domain
- **Automatic** cert issuance through CertBot (by EFF) on web server
- **Secure**: “LE will serve as a platform for advancing TLS security...”
- **Transparent**: All certs issued and revoked are publicly logged
- **Open**: Cert management process is published as open source software.
- **Cooperative**: Joint effort between multiple organizations and community

Who made this happen?
I want to see the proof!

- **Linux Foundation**
- Sponsored by many large organizations
  - Mozilla, Cisco, EFF, Google Chrome, Facebook, SquareSpace, Shopify, Hewlett Packard…
  - Many more

How does it work (Root Cert Propagation)

- **LE Root Certificate** (ISRG Root 1X)
  - Kept safely offline
  - Propagated through Intermediates
- **LE Intermediate Certificates** (All IdentTrust cross-signed)
  - X1, X2 - Original Intermediates
  - X3 - Current generation Intermediate
  - X4 - Disaster Recovery Intermediate



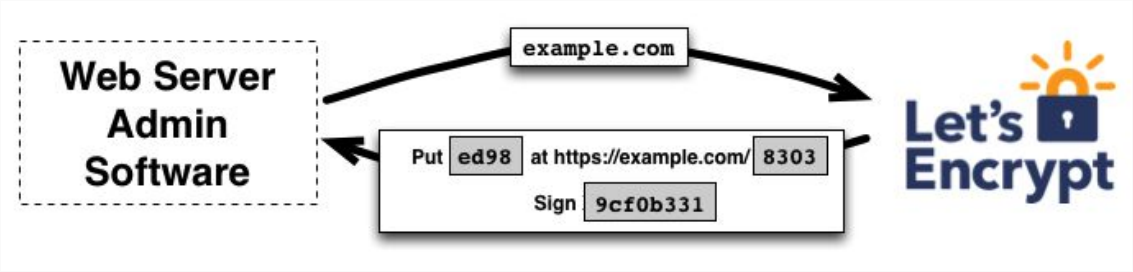
Crazy Diagram!

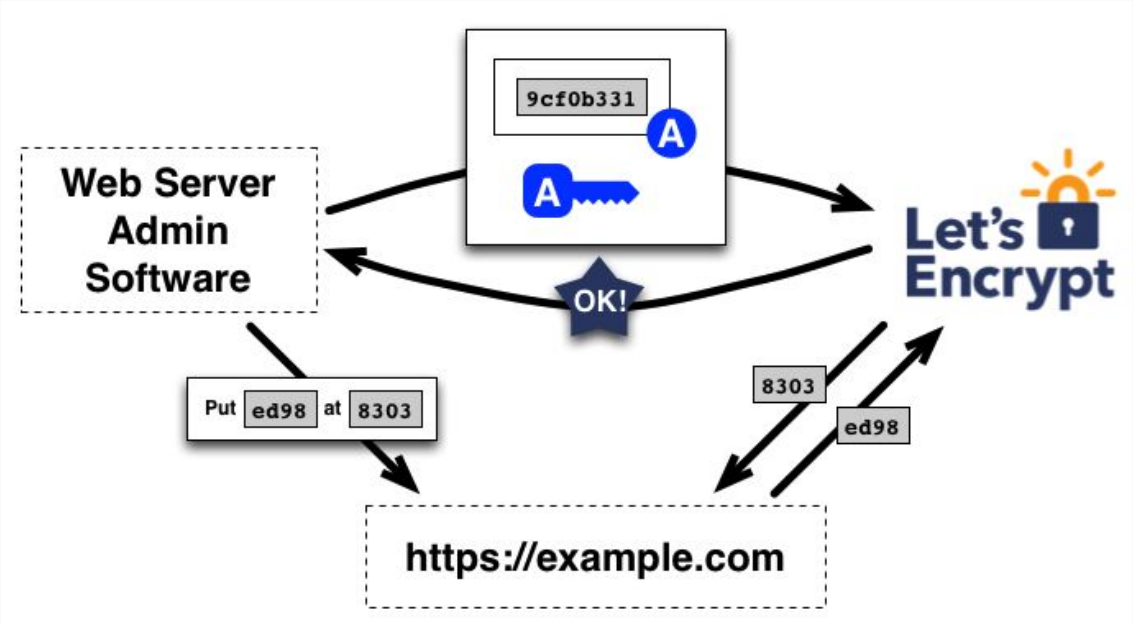
How does it work? (Domain Verification)

- Automatic verification via DNS
- Three modes
  - **Webroot**: Domain verification service looks for file in the public web directory
  - **Standalone**: Uses ports 80/443 to respond to request from domain verification service
  - **Automatic**: Plugins for Apache and nginx
- Uses URL / key pairs

Verification Process

- Challenge Sets
  - Adding key to a specific, random URL
  - Verify from LE servers





Getting your certificates

Installation (*Certbot*)

- Nowadays, available in most Linux package repositories
  - *If not*: Compile from source and run it (all Python underneath)
- Debian / Ubuntu / Debian-based distributions
  - `$ sudo apt-get install certbot`
- Red Hat Enterprise Linux / CentOS (via EPEL)
  - `$ sudo yum install certbot`
- Fedora
  - `$ sudo dnf install certbot`
- Arch Linux
  - `$ sudo pacman -S certbot`

Issuing certificates: Webroot method

- Webroot uses root directory of your domain to verify domain authenticity
  - Places files in root directory, LE servers check if files are present
  - Most useful when using a CDN or something else in between connections to your servers
- Run the following command to get your certificate(s):

```
$ sudo certbot certonly -m me@example.com --webroot -w
/var/www/example.com/public_html/ -d example.com
```

Issuing certificates: Standlone method

- Standalone uses port 80 / 443 to verify domain authenticity
  - Requires ports 80 or 443 to not already be in use
- Run the following command to get your certificate(s):

```
$ sudo certbot certonly -m me@example.com --standalone -d
example.com --pre-hook=”systemctl stop nginx”
--post-hook=”systemctl start nginx”
```

Renewing certificates

- Renewing your certificates is… actually easy
- Run the following command to get your certificate(s):

```
$ sudo certbot renew
```

Run it in prod!

Writing an nginx conf for *ex.io* (1/3)

```
server {
    listen        443 ssl;
    server_name   ex.io;
    root          /var/www/ex.io/public_html;

    access_log    /var/www/ex.io/logs/ex.io_access.log;
    error_log     /var/www/ex.io/logs/ex.io_error.log error;
```

Writing an nginx conf for *ex.io* (2/3)

```
    ssl                         on;
    ssl_certificate
/etc/ssl/certs/ex_io/ex_io-fullchain.pem;
    ssl_certificate_key
/etc/ssl/certs/ex_io/ex_io-privkey.pem;
    ssl_protocols               TLSv1 TLSv1.1 TLSv1.2;
    ssl_ciphers
"SSLv3:TLSv1:+HIGH:!SSLv2:!MD5:!MEDIUM:!LOW:!EXP:!ADH:!eNU
LL:!aNULL";
    ssl_prefer_server_ciphers on;
```

Writing an nginx conf for *ex.io* (3/3)

```
    location / {
        index  index.html index.htm;
        server_tokens off;
    }
}
server {
    listen       80;
    server_name ex.io;
    rewrite      ^   https://$server_name$request_uri?
permanent;
}
```

Just like that!

Developer Tools - https://serubin.net/

Certificate Viewer: serubin.net

General | Details

This certificate has been verified for the following usages:

SSL Server Certificate

Issued To

Common Name (CN)	serubin.net
Organization (O)	<Not Part Of Certificate>
Organizational Unit (OU)	<Not Part Of Certificate>
Serial Number	03:11:FE:FF:98:33:FF:CE:8A:E5:8E:D6:2E:13:79:EF:CD

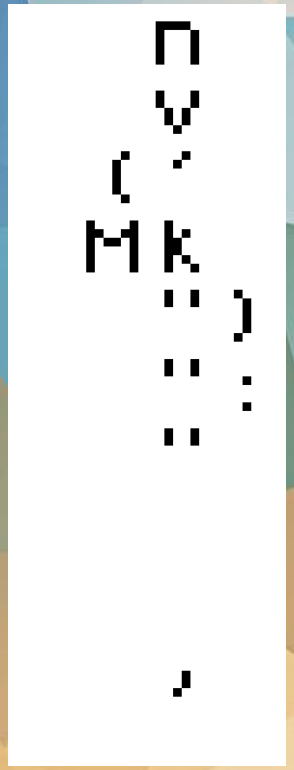
Issued By

Common Name (CN)	Let's Encrypt Authority X3
Organization (O)	Let's Encrypt
Organizational Unit (OU)	<Not Part Of Certificate>

Live Demo: nginx

Completely and totally unrehearsed.

brokenencryptionmakesmecry.jwf.io



Questions?
Comments?
Suggestions?

- Justin W. Flory
- Solomon Rubin