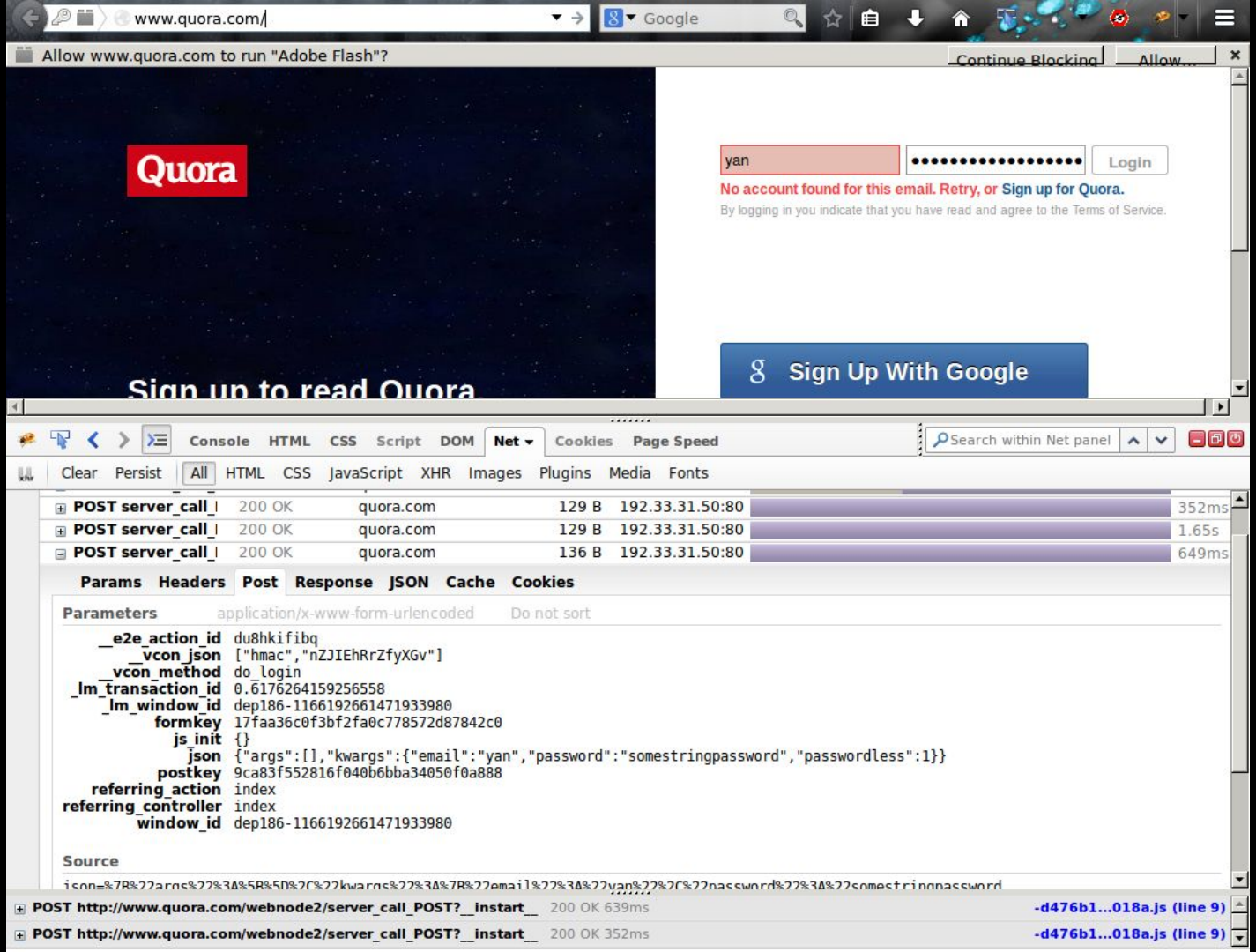
Horror Stories about the Encrypted Web

(and how Let's Encrypt is helping)

{pde,yan}@eff.org

Horror Story #1

We don’t live in a 100% HTTPS world


www.quora.com/
Google
Allow www.quora.com to run "Adobe Flash"?
Continue Blocking
Allow...
Quora
yan
Login
No account found for this email. Retry, or Sign up for Quora.
By logging in you indicate that you have read and agree to the Terms of Service.
Sign Up With Google
Console HTML CSS Script DOM Net Cookies Page Speed
Search within Net panel
Clear Persist All HTML CSS JavaScript XHR Images Plugins Media Fonts
POST server_call_ 200 OK quora.com 129 B 192.33.31.50:80 352ms
POST server_call_ 200 OK quora.com 129 B 192.33.31.50:80 1.65s
POST server_call_ 200 OK quora.com 136 B 192.33.31.50:80 649ms
Params Headers Post Response JSON Cache Cookies
Parameters application/x-www-form-urlencoded Do not sort
__e2e_action_id du8hkifibq
__vcon_json ["hmac","nZJIEhRrZfyXGv"]
__vcon_method do_login
_lm_transaction_id 0.6176264159256558
_lm_window_id dep186-1166192661471933980
formkey 17faa36c0f3bf2fa0c778572d87842c0
js_init {}
json {"args":[],"kwargs":{"email":"yan","password":"somestringpassword","passwordless":1}}
postkey 9ca83f552816f040b6bba34050f0a888
referring_action index
referring_controller index
window_id dep186-1166192661471933980
Source
POST http://www.quora.com/webnode2/server_call_POST?__instart__ 200 OK 639ms -d476b1...018a.js (line 9)
POST http://www.quora.com/webnode2/server_call_POST?__instart__ 200 OK 352ms -d476b1...018a.js (line 9)

June, 2014

Horror Story #2

Setting up TLS is tedious, even in 2016

Purchasing a Signed Certificate from a Certificate Authority (CA)

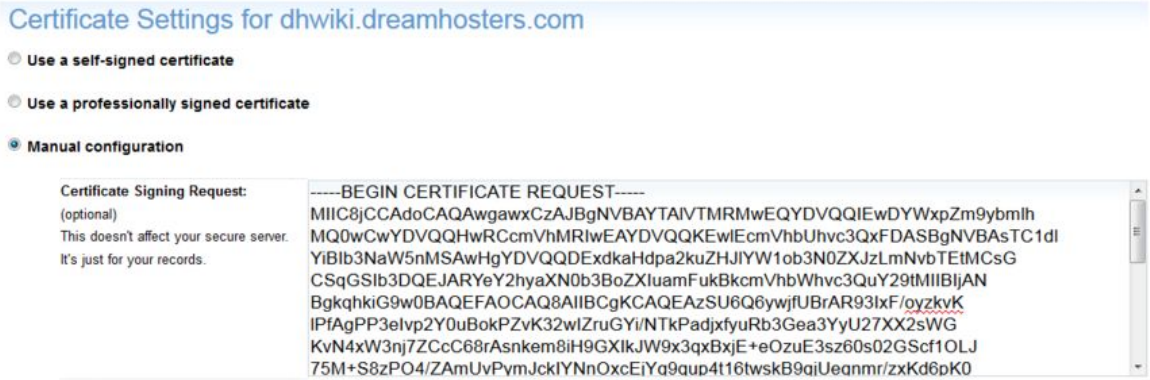
You can also purchase a certificate directly from a Certificate Authority (CA) and install it in to your DreamHost panel. To do this, you'll need a Certificate Signing Request (CSR) which can be found in your panel.

The following steps explain how to obtain this CSR in your panel:

1. Review the Adding Secure Hosting (self-signed certificate) section above to add Secure Hosting and a self-signed certificate to your domain.
2. Go to the (Panel > 'Domains' > 'Manage Domains') page.

   The 'Manage Domains' page opens:
3. To the right of your domain, click on the Certificates button.
4. When the 'Secure Hosting' page opens, click the 'Manual configuration' radio button to expose the current certificate information.

   There are several large text fields on this page:



5. COPY (do NOT cut) the text from the Certificate Signing Request field box.
6. Paste the text into the order form from whichever Certificate Authority you'd like to purchase your signed SSL certificate.

Important:

When purchasing a signed SSL certificate, you must specify the server type:

- To use the SSL certificate on DreamHost's servers, specify the server type as: `Apache 2.X w/MOD_SSL`
- Once you successfully complete your purchase, the CA will send you the signed SSL certificate; you can then replace the self-signed SSL certificate with this signed SSL certificate in your panel.
- For more information, see the instructions in the following Installing a Sign Certificate you've already purchased section.

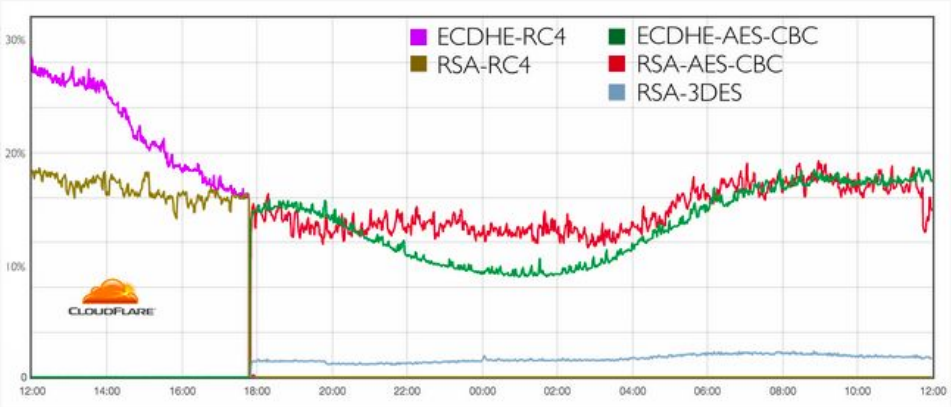
Horror Story #3

TLS configuration is confusing

It is widely believed that AES-CBC is a secure cipher for the long term, unlike RC4. Choosing AES-CBC provides our customers with long-term forward secrecy, even if it could open them up to a rarely executed noisy active attack if they are using an out of date browser and OS. Choosing RC4 exposes our customers' data to any government who has advanced enough cryptographic techniques to break it. When faced with this choice, we would rather protect our customers from long term threats by choosing AES-CBC. Experts agree, it's time to move on from RC4.

Who uses RC4?

Instead of just removing RC4 altogether, we decided first to lower the priority of the RC4 cipher suites on our servers. Typically in HTTPS, the client lets the server know which cipher suites it supports, from which the server picks their favorite. By putting RC4 last in terms of server preference, we will only choose RC4 if it the client does not support anything else. The following chart shows what happened when we changed our cipher preferences.




FEAR CAN HOLD YOU PRISONER.
HOPE CAN SET YOU FREE.
TIM ROBBINS MORGAN FREEMAN
THE
SHA - 256
REDEMPTION
CASTLE ROCK ENTERTAINMENT Presents
A FRANK DARABONT film TIM ROBBINS MORGAN FREEMAN "THE SHAWSHANK REDEMPTION" BOB GUNTON WILLIAM SADLER
CLANCY BROWN GIL BELLOWS AND JAMES WHITMORE AS "BROOKS" Music by THOMAS NEWMAN Edited by RICHARD FRANCIS-BRUCE
Production Designer TERENCE MARSH Director of Photography ROGER DEAKINS, B.S.C. Executive Producers LIZ GLOTZER AND DAVID LESTER Based on the Short Novel Rita Hayworth and Shawshank Redemption by STEPHEN KING
Screenplay by FRANK DARABONT Produced by NIKI MARVIN Directed by FRANK DARABONT
CASTLE ROCK ENTERTAINMENT
COLUMBIA PICTURES
THIS FALL

```
ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
ssl_prefer_server_ciphers on;


# Using list of ciphers from "Bulletproof SSL and TLS"
ssl_ciphers "ECDHE-ECDSA-AES128-GCM-SHA256 ECDHE-ECDSA-
AES256-GCM-SHA384 ECDHE-ECDSA-AES128-SHA ECDHE-ECDSA-
AES256-SHA ECDHE-ECDSA-AES128-SHA256 ECDHE-ECDSA-
AES256-SHA384 ECDHE-RSA-AES128-GCM-SHA256 ECDHE-RSA-
AES256-GCM-SHA384 ECDHE-RSA-AES128-SHA ECDHE-RSA-
AES128-SHA256 ECDHE-RSA-AES256-SHA384 DHE-RSA-AES128-
GCM-SHA256 DHE-RSA-AES256-GCM-SHA384 DHE-RSA-AES128-
SHA DHE-RSA-AES256-SHA DHE-RSA-AES128-SHA256 DHE-RSA-
AES256-SHA256 EDH-RSA-DES-CBC3-SHA";
```

Horror Story #4

Mixed content blocking

https://www.lenovo.com/us/en/

Lenovo

PRODUCTS DEALS SUPPORT

Sales 1-855-253-6686 Email Sign-Up

For Work Account Search

WINDOWS 10 IS HERE

[–] Feedback

Elements Network Sources Timeline Profiles Resources Audits Console HTTPS Everywhere 202 2

<top frame> Preserve log

Mixed Content: The page at 'https://www.lenovo.com/us/en/' was loaded over HTTPS, but requested an insecure font 'http://www.lenovo.com/au/en/fonts/lenovobd-webfont.woff'. This request has been blocked; the content must be served over HTTPS. (index):1

Mixed Content: The page at 'https://www.lenovo.com/us/en/' was loaded over HTTPS, but requested an insecure font 'http://www.lenovo.com/au/en/fonts/lenovobd-webfont.ttf'. This request has been blocked; the content must be served over HTTPS. (index):1

Mixed Content: The page at 'https://www.lenovo.com/us/en/' was loaded over HTTPS, but requested an insecure font 'http://www.lenovo.com/au/en/fonts/icomoon.woff'. This request has been blocked; the content must be served over HTTPS. (index):1

Mixed Content: The page at 'https://www.lenovo.com/us/en/' was loaded over HTTPS, but requested an insecure font 'http://www.lenovo.com/au/en/fonts/icomoon.ttf'. This request has been blocked; the content must be served over HTTPS. (index):1

Mixed Content: The page at 'https://www.lenovo.com/us/en/' was loaded over HTTPS, but requested an insecure font 'http://www.lenovo.com/au/en/fonts/lenovolg-webfont.woff'. This request has been blocked; the content must be served over HTTPS. (index):1

Mixed Content: The page at 'https://www.lenovo.com/us/en/' was loaded over HTTPS, but requested an insecure font 'http://www.lenovo.com/au/en/fonts/lenovolg-webfont.ttf'. This request has been blocked; the content must be served over HTTPS. (index):1

Mixed Content: The page at 'https://www.lenovo.com/us/en/' was loaded over HTTPS, but requested an insecure font 'http://www.lenovo.com/au/en/fonts/lenovorg-webfont.woff'. This request has been blocked; the content must be served over HTTPS. (index):1

Mixed Content: The page at 'https://www.lenovo.com/us/en/' was loaded over HTTPS, but requested an insecure font 'http://www.lenovo.com/au/en/fonts/lenovorg-webfont.ttf'. This request has been blocked; the content must be served over HTTPS. (index):1

Mixed Content: The page at 'https://www.lenovo.com/us/en/' was loaded over HTTPS, but requested an insecure font 'http://www.lenovo.com/au/en/fonts/lenovomd-webfont.woff'. This request has been blocked; the content must be served over HTTPS. (index):1

Mixed Content: The page at 'https://www.lenovo.com/us/en/' was loaded over HTTPS, but requested an insecure font 'http://www.lenovo.com/au/en/fonts/lenovomd-webfont.ttf'. This request has been blocked; the content must be served over HTTPS. (index):1

Mixed Content: The page at 'https://www.lenovo.com/us/en/' was loaded over HTTPS, but requested an insecure font 'http://www.lenovo.com/au/en/fonts/lenovobd-webfont.woff'. This request has been blocked; the content must be served over HTTPS. (index):1

Mixed Content: The page at 'https://www.lenovo.com/us/en/' was loaded over HTTPS, but requested an insecure font 'http://www.lenovo.com/au/en/fonts/lenovobd-webfont.ttf'. This request has been blocked; the content must be served over HTTPS. (index):1

§ 1.2. Examples

EXAMPLE 1

Megacorp, Inc. wishes to migrate `http://example.com/` to `https://example.com`. They set up their servers to make their own resources available over HTTPS, and work with partners in order to make third-party widgets available securely as well.

They quickly realize, however, that the majority of their content is locked up in a database tied to an old content management system, and it contains hardcoded links to insecure resources (e.g., http:// URLs to images and other content). Unfortunately, it's a substantial amount of work to update it.

As a stopgap measure, Megacorp injects the following header field into every HTML response that goes out from their servers:

```
Content-Security-Policy: upgrade-insecure-requests
```

This automatically upgrades all insecure resource requests from their pages to secure variants, allowing a user agent to treat the following HTML code:

```
<img src="http://example.com/image.png">
<img src="http://not-example.com/image.png">
```

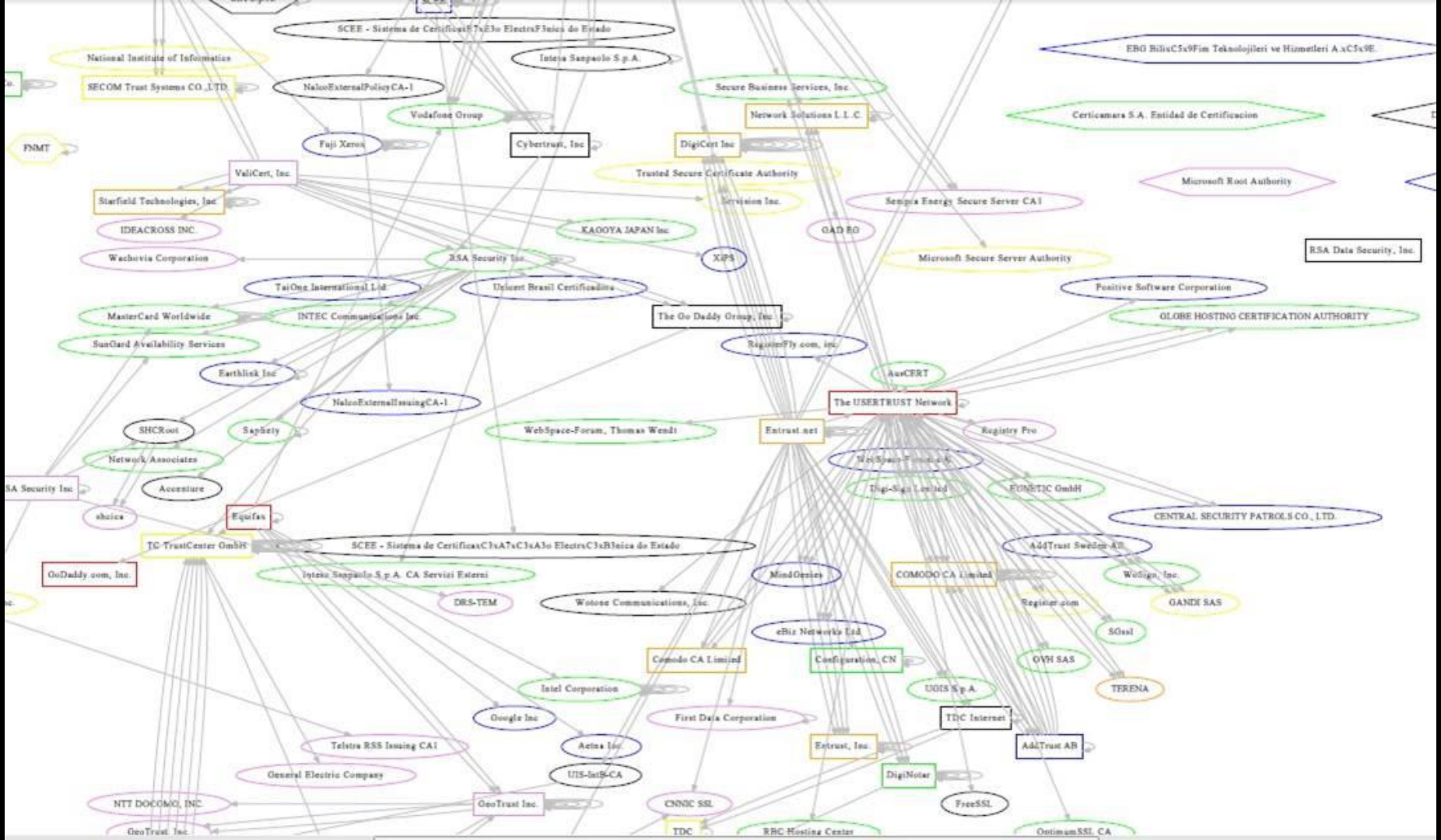
as though it had been delivered as:

```
<img src="https://example.com/image.png">
<img src="https://not-example.com/image.png">
```

```
Content Security Policy: Upgrading insecure request 'http://zyan.scripts.mit.edu/blog/wp-content/uploads/2015/08/Screen-Shot-2015-08-23-at-1.35.48-PM.png' to use 'https'  <unknown>
Content Security Policy: Upgrading insecure request 'http://zyan.scripts.mit.edu/blog/wp-content/uploads/2015/08/Screen-Shot-2015-08-23-at-1.39.47-PM.png' to use 'https'  <unknown>
Content Security Policy: Upgrading insecure request 'http://zyan.scripts.mit.edu/blog/wp-content/uploads/2015/08/Screen-Shot-2015-08-23-at-4.44.45-PM.png' to use 'https'  <unknown>
Content Security Policy: Upgrading insecure request 'http://zyan.scripts.mit.edu/blog/wp-content/uploads/2015/08/Screen-Shot-2015-08-23-at-4.45.10-PM.png' to use 'https'  <unknown>
Content Security Policy: Upgrading insecure request 'http://zyan.scripts.mit.edu/blog/wp-content/uploads/2015/08/Screen-Shot-2015-08-23-at-7.48.14-PM.png' to use 'https'  <unknown>
Content Security Policy: Upgrading insecure request 'http://zyan.scripts.mit.edu/blog/wp-content/uploads/2015/08/Screen-Shot-2015-08-23-at-4.43.10-PM.png' to use 'https'  <unknown>
Content Security Policy: Upgrading insecure request 'http://zyan.scripts.mit.edu/blog/wp-content/uploads/2015/08/Screen-Shot-2015-08-23-at-4.43.42-PM.png' to use 'https'  <unknown>
Content Security Policy: Upgrading insecure request 'http://zyan.scripts.mit.edu/blog/wp-content/uploads/2015/08/defcon1-624x624.jpg' to use 'https'  <unknown>
Content Security Policy: Upgrading insecure request 'http://zyan.scripts.mit.edu/blog/wp-content/uploads/2015/08/defcon2-624x624.jpg' to use 'https'  <unknown>
Content Security Policy: Upgrading insecure request 'http://zyan.scripts.mit.edu/blog/wp-content/uploads/2015/08/defcon3-624x624.jpg' to use 'https'  <unknown>
Content Security Policy: Upgrading insecure request 'http://zyan.scripts.mit.edu/blog/wp-content/uploads/2015/08/silentcircle-624x624.jpg' to use 'https'  <unknown>
Content Security Policy: Upgrading insecure request 'http://zyan.scripts.mit.edu/blog/wp-content/uploads/2015/08/defcon5-624x624.jpg' to use 'https'  <unknown>
Content Security Policy: Upgrading insecure request 'http://zyan.scripts.mit.edu/blog/wp-content/uploads/2015/08/pocorgtfo-624x624.jpg' to use 'https'  <unknown>
Content Security Policy: Upgrading insecure request 'http://zyan.scripts.mit.edu/blog/wp-content/uploads/2015/07/Screen-Shot-2015-07-26-at-2.07.14-PM.png' to use 'https'  <unknown>
Content Security Policy: Upgrading insecure request 'http://zyan.scripts.mit.edu/blog/wp-content/uploads/2015/07/Screen-Shot-2015-07-26-at-3.02.59-PM.png' to use 'https'  <unknown>
```

Horror Story #5

There are too many certificate authorities.


SCEE - Sistema de Certificaci?n Electr?nica do Estado
National Institute of Informatics
Intesa Sanpaolo S.p.A.
EBG Bili?im Teknolojileri ve Hizmetleri A.?.
SECOM Trust Systems CO.,LTD.
NaiconExternalPolicyCA-1
Secure Business Services, Inc.
Vodafone Group
Network Solutions L.L.C.
Certicamara S.A. Entidad de Certificacion
FNMT
Fuji Xerox
Cybertrust, Inc
DigiCert Inc
ValiCert, Inc.
Trusted Secure Certificate Authority
Microsoft Root Authority
Starfield Technologies, Inc.
Verisign Inc.
Sempra Energy Secure Server CA1
IDEACROSS INC.
XAOOYA JAPAN Inc
GAD EG
Wachovia Corporation
RSA Security Inc.
XiPS
Microsoft Secure Server Authority
RSA Data Security, Inc.
TaiOne International Ltd
Unicert Brasil Certificadora
Positive Software Corporation
MasterCard Worldwide
INTEC Communications Inc
The Go Daddy Group, Inc.
GLOBE HOSTING CERTIFICATION AUTHORITY
SunGard Availability Services
RegisterFly.com, inc.
Earthlink Inc
AusCERT
NaiconExternalIssuingCA-1
The USERTRUST Network
SHCRoot
Saphety
WebSpace-Forum, Thomas Wendt
Entrust.net
Registry Pro
Network Associates
Digi-Sign Limited
FUNSTIC GmbH
Accenture
Equifax
CENTRAL SECURITY PATROLS CO., LTD.
TC TrustCenter GmbH
SCEE - Sistema de Certificaci?n Electr?nica do Estado
AddTrust Sweden AB
GoDaddy.com, Inc.
Intesa Sanpaolo S.p.A. CA Servizi Esterni
MindGeeks
COMODO CA Limited
Wedigo, Inc.
DRS-TEM
Watone Communications, Inc.
Register.com
GANDI SAS
eBiz Networks Ltd
SGssl
Comodo CA Limited
Configuration, CN
OVH SAS
Intel Corporation
UGIS S.p.A.
TERENA
Google Inc
First Data Corporation
TDC Internet
Telstra RSS Issuing CA1
Aetna Inc
Entrust, Inc.
AddTrust AB
General Electric Company
UIS-IntS-CA
DigiNotar
NTT DOCOMO, INC.
GeoTrust Inc.
CNNIC SSL
FreeSSL
TDC
RBC Hosting Center
OptimumSSL CA

it’s time to fight back

So we started a CA...

(one more CA)

Let's Encrypt is a new Certificate Authority:

It's free, automated, and open.

In Public Beta

Let's Encrypt created by

- Engineering: EFF, Mozilla, University of Michigan
- Financial sponsorship: Cisco, Akamai
- CA cross-signature: IdentTrust
- Housed in a new 501(c)3, the Internet Security Research Group (ISRG)

Platinum

Gold

Silver

Security

How do we decide whether to issue a cert?

Dialog: ACME protocol

Shrubberies: ACME “challenges”

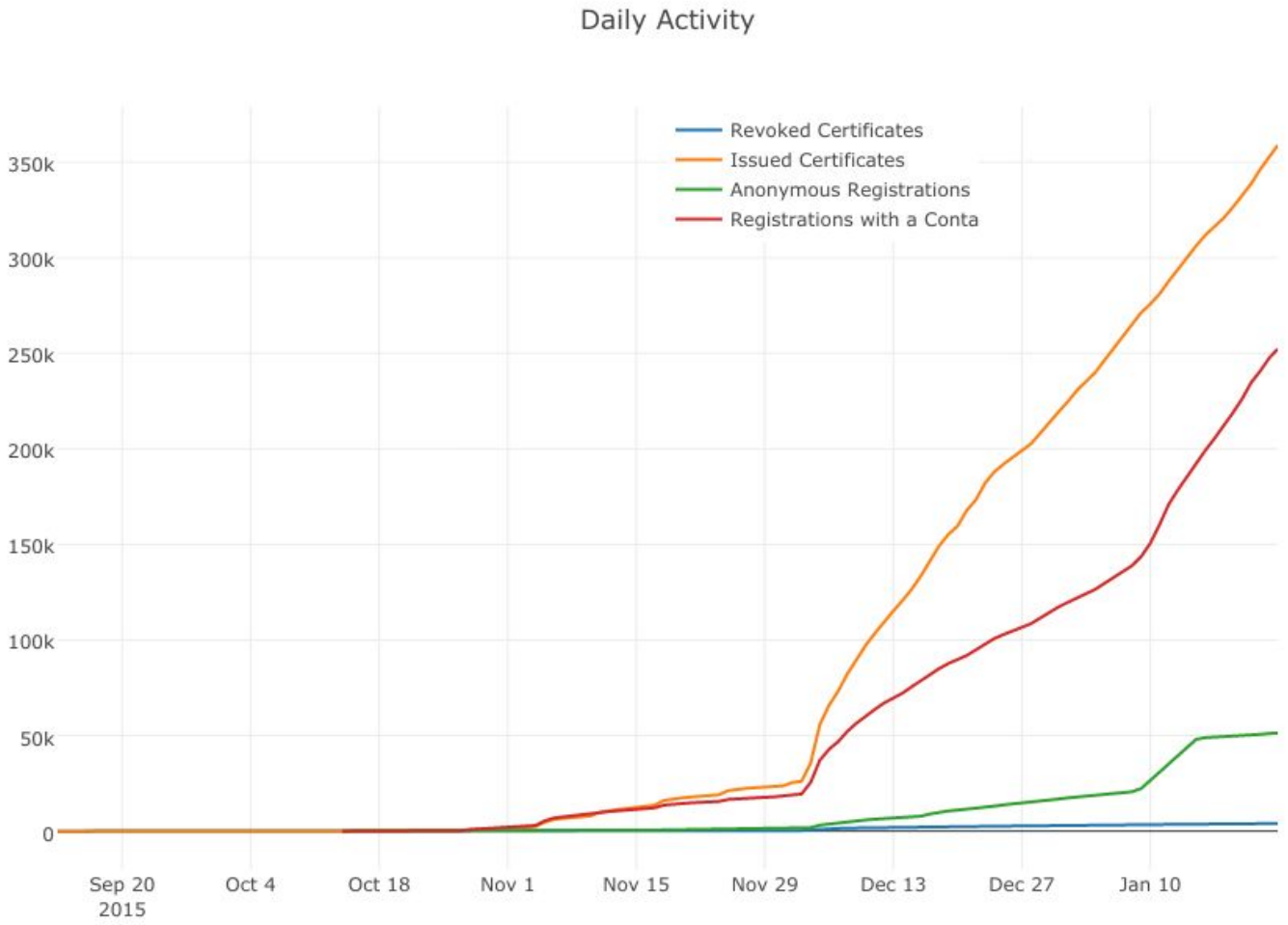
Current status

Private beta through Nov, 2015

Entered public beta on Dec. 3, 2015

Issued 10k certs in <8 hours (1 cert / 3 seconds!)

Almost 400k certs issued so far!


Daily Activity
Revoked Certificates
Issued Certificates
Anonymous Registrations
Registrations with a Conta
350k
300k
250k
200k
150k
100k
50k
0
Sep 20
2015
Oct 4
Oct 18
Nov 1
Nov 15
Nov 29
Dec 13
Dec 27
Jan 10

More statistics

374714 certificates checked (totalling 801637 DNS names)

adoption statistics

names using issued cert	**547,200**	(68.26%)
certs used by all names	162,844	(43.46%)
certs used by some names	11,341	(3.03%)
certs used by no names	200,529	(53.52%)

```
# cipher suite breakdown

 ECDHE RSA WITH AES 256 CBC SHA      4,817     (1.33%)
 RSA WITH AES 256 CBC SHA            1,354     (0.37%)
 RSA WITH AES 128 CBC SHA            27,551    (7.63%)
 RSA WITH 3DES EDE CBC SHA           104       (0.03%)
 ECDHE RSA WITH 3DES EDE CBC SHA     30        (0.01%)
 ECDHE RSA WITH AES 128 GCM SHA256   222,517   (61.59%)
 ECDHE RSA WITH AES 256 GCM SHA384   98,427    (27.24%)
 ECDHE RSA WITH AES 128 CBC SHA      6,516     (1.80%)
```

Lots of forward secrecy!

Alexa top domains using Let's Encrypt

- archlinux.org
- teamliquid.net (Starcraft news site)
- overclockers.ru (electronics / tech news site)
- gimp.org
- distrowatch.com
- goodlife.tw (shopping promotions site)
- douglas.de (cosmetics site)
- More at https://censys.io/domain?q=%28*%29+AND+443.https.tls.certificate.parsed.issuer.common_name%3A+%22Let%27s+Encrypt+Authority+X1%22

Client types and plans...

Large, custom infrastructures

Multi-server (load balanced)

Single server (VPS, self-hosted, managed hosting etc)

Bulk hosting (no user shell)

Large, custom infrastructures

Multi-server (load balanced)

Single server (VPS, self-hosted, managed hosting etc)

Bulk hosting (no user shell)

Large, custom infrastructures

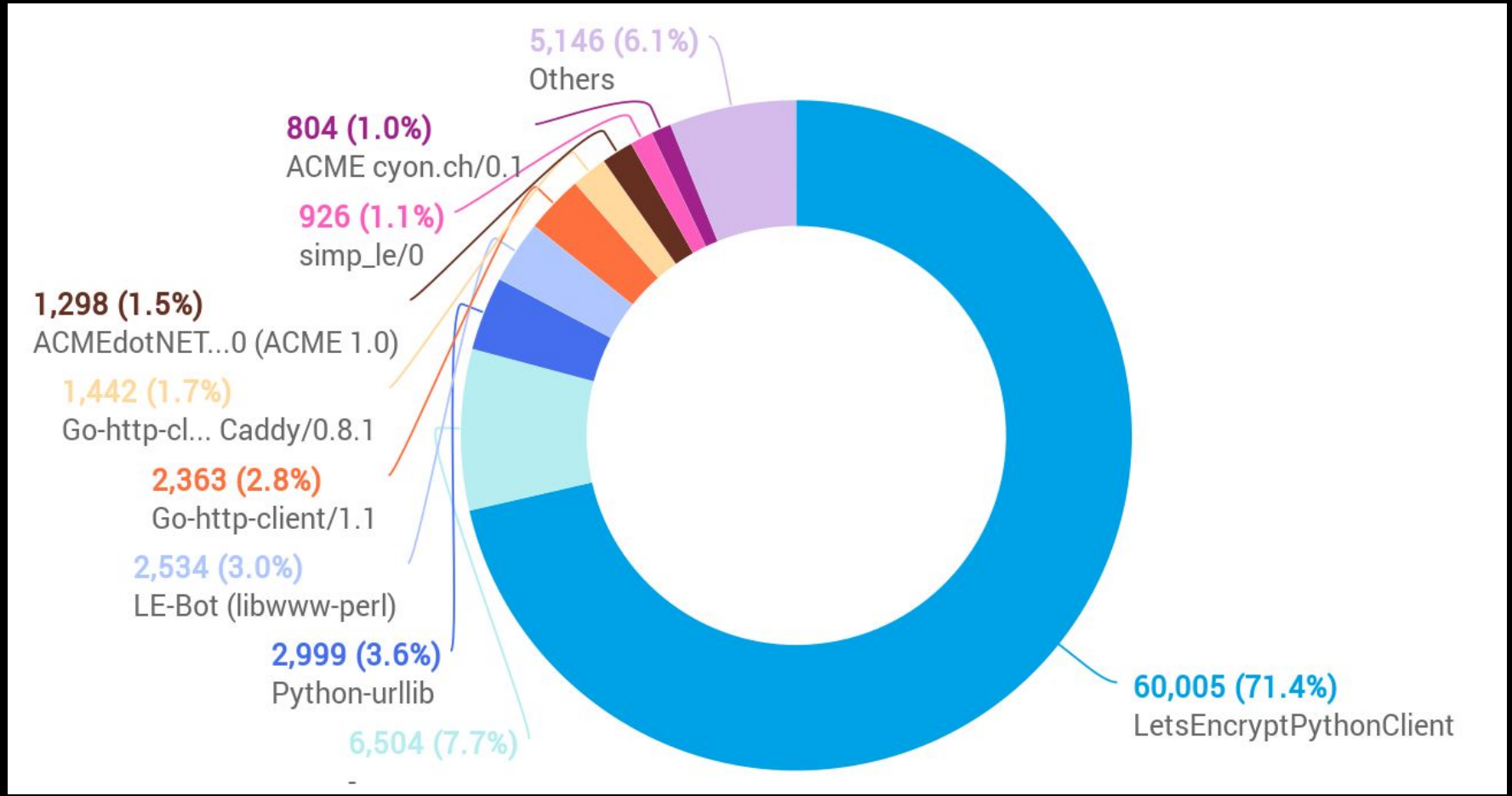
Multi-server (load balanced)

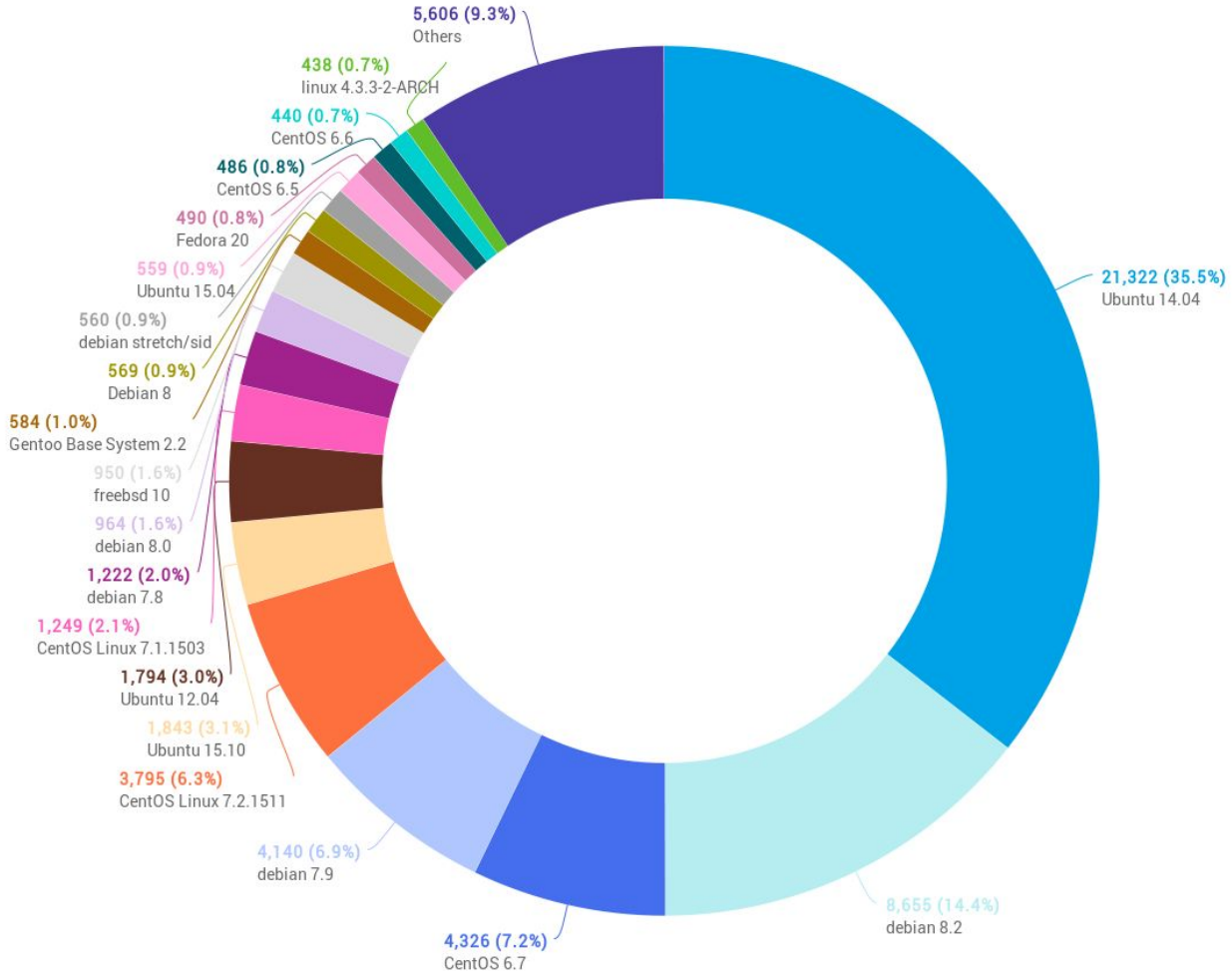
Single server (VPS, self-hosted, managed hosting etc)

Bulk hosting (no user shell)


Large, custom infrastructures
Multi-server (load balanced)
Single server (VPS, self-hosted, managed hosting etc)
Bulk hosting (no user shell)

Diverse clients...


5,146 (6.1%)
Others
804 (1.0%)
ACME cyon.ch/0.1
926 (1.1%)
simp_le/0
1,298 (1.5%)
ACMEdotNET...0 (ACME 1.0)
1,442 (1.7%)
Go-http-cl... Caddy/0.8.1
2,363 (2.8%)
Go-http-client/1.1
2,534 (3.0%)
LE-Bot (libwww-perl)
2,999 (3.6%)
Python-urllib
6,504 (7.7%)
-
60,005 (71.4%)
LetsEncryptPythonClient


5,606 (9.3%)
Others
438 (0.7%)
linux 4.3.3-2-ARCH
440 (0.7%)
CentOS 6.6
486 (0.8%)
CentOS 6.5
490 (0.8%)
Fedora 20
559 (0.9%)
Ubuntu 15.04
560 (0.9%)
debian stretch/sid
569 (0.9%)
Debian 8
584 (1.0%)
Gentoo Base System 2.2
950 (1.6%)
freebsd 10
964 (1.6%)
debian 8.0
1,222 (2.0%)
debian 7.8
1,249 (2.1%)
CentOS Linux 7.1.1503
1,794 (3.0%)
Ubuntu 12.04
1,843 (3.1%)
Ubuntu 15.10
3,795 (6.3%)
CentOS Linux 7.2.1511
4,140 (6.9%)
debian 7.9
4,326 (7.2%)
CentOS 6.7
8,655 (14.4%)
debian 8.2
21,322 (35.5%)
Ubuntu 14.04

Rat

https://github.com/letsencrypt/letsencrypt/blob/master/letsencrypt/interfaces.py#L132

```
class IAuthenticator(IPlugin):
    """Generic Let's Encrypt Authenticator."""

    def get_chall_pref(domain):
        """Return list of challenge preferences."""

    def perform(achalls):
        """Perform the given challenge."""

    def cleanup(achalls):
        """Revert changes and shutdown after challenges complete."""
```

https://github.com/letsencrypt/letsencrypt/blob/master/letsencrypt/interfaces.py#L230

```
class IInstaller(IPlugin):
    """Generic Let's Encrypt Installer Interface. """

    def get_all_names():
        """Returns all names that may be authenticated."""

    def deploy_cert(domain, cert_path, key_path, chain_path, fullchain_path):
        """Deploy certificate."""

    def enhance(domain, enhancement, options=None):
        """Perform a configuration enhancement.  """

    def supported_enhancements():
        """Returns a list of supported enhancements.  """

    def get_all_certs_keys():
        """Retrieve all certs and keys set in configuration.  """

    def save(title=None, temporary=False):
        """Saves all changes to the configuration files.  """

    def rollback_checkpoints(rollback=1):
        """Revert `rollback` number of configuration checkpoints.  """

    def recovery_routine():
        """Revert configuration to most recent finalized checkpoint. """

    def view_config_changes():
        """Display all of the LE config changes."""

    def config_test():
        """Make sure the configuration is valid."""

    def restart():
        """Restart or refresh the server content."""
```

Plugins used on Ubuntu 14:

Authenticator/standalone	Installer/none	6,602
Authenticator/webroot	Installer/none	5,769
Authenticator/apache	Installer/apache	4,875
Authenticator/plesk	Installer/plesk	2,850
Authenticator/manual	Installer/none	674
Authenticator/apache	Installer/none	351
Authenticator/webroot	Installer/apache	92
Authenticator/standalone	Installer/apache	56
Authenticator/nginx	Installer/nginx	29
Authenticator/manual	Installer/apache	28
Authenticator/s3front	Installer/s3front	23
Authenticator/nginx	Installer/none	13
Authenticator/webroot	Installer/nginx	2
Authenticator/standalone	Installer/null	1
Authenticator/gandi-shs	Installer/gandi-shs	1
Authenticator/gandi-shs	Installer/none	1
Authenticator/webroot	Installer/s3front	1
Authenticator/manual	Installer/nginx	1

How well configured are we?

```
# some further work required

    # name problems

     invalid DNS             12106   (1.51%)
     refused/unavailable     27108   (3.38%)
     timed out               22628   (2.82%)
     TLS error               7627    (0.95%)
     sent incomplete chain   26648   (3.32%)
     expired cert            5582    (0.70%)
     self-signed cert        10      (0.00%)
     cert has wrong names    84172   (10.50%)
     misc. invalid cert      3       (0.00%)

    # feature usage

     OCSP stapled   52797   (6.59%)
     SCT included   159     (0.02%)
```

Vulnerability reporting

1 Open ✓ 10 Closed — Author ▾ Labels ▾ Milestones ▾ Assignee ▾ Sort ▾

Test that our copy of go-jose ignores the JWS "algorithm" field sec-high tests
#1081 opened on Nov 3, 2015 by pde — 0

Remove insecure challenges bug pr-in-progress sec-high spec-compliance
#894 opened on Oct 1, 2015 by bifurcation — Sprint 2015-12-09 — 8

Address signature reuse vulnerability ✓ r=jcjones r=jmhodges r? sec-high spec-compliance
#774 opened on Sep 10, 2015 by bifurcation — General Availability — 69

Mitigate signature misuse vulnerability bug sec-high
#604 opened on Aug 11, 2015 by jsha — Deployment — 1

SimpleHTTP validation accepts ".." components bug sec-high
#313 opened on Jun 5, 2015 by jcjones — Deployment — 0

Do not accept HMAC-based JWS signatures enhancement sec-high
#259 opened on May 29, 2015 by bifurcation — Sprint 2015-11-03 — 1

Verify safety of using "encoding/asn1" parsing (e.g. of CSRs) question sec-high
#246 opened on May 27, 2015 by bifurcation — Deployment — 6

WFE should not return arbitrary errors to client bug sec-high
#174 opened on May 11, 2015 by jsha — Deployment — 1

Some endpoints in web-front-end.go need to check for key continuity bug sec-high
#75 opened on Apr 1, 2015 by diracdeltas — Deployment — 4

Audit Logging audit-compliance enhancement sec-high
#62 opened on Mar 25, 2015 by jcjones — Deployment — 4

Don't ignore random number generation errors sec-high
#51 opened on Mar 22, 2015 by diracdeltas — Deployment — 3

[Acme] Signature misuse vulnerability in draft-barnes-acme-04

Andrew Ayer <agwa@andrewayer.name> | Tue, 11 August 2015 15:54 UTC | Show header

I recently reviewed draft-barnes-acme-04 and found vulnerabilities in the DNS, DVSNI, and Simple HTTP challenges that would allow an attacker to fraudulently complete these challenges.

I shall describe the DNS challenge vulnerability first, since it is the most serious as it requires no MitM or other network-layer subversion. The assumptions are:

Mallory wants to prove ownership of example.com via DNS challenge

1. Mallory registers RSA key pair [1] for challenge signing
2. letsencrypt issues DNS challenge [2]
3. Mallory queries example.com's TXT record [3] from when example.com solved its own challenge.
4. **Mallory constructs a new RSA key pair [4] such that [3] is a valid signature over [2].**
5. Mallory uses letsencrypt account recovery process to replace [1] with [4].
6. letsencrypt verifies that [3] is a valid signature from Mallory's new account key, and issues Mallory cert for example.com.


gifbin.com

“The real problem is that ACME makes false assumptions about signatures. It assumes that a signature uniquely identifies a (public key, message) tuple, which RSA does not guarantee.”

Address signature reuse vulnerability #774

Merged **jmhodges** merged 42 commits into `master` from `sig-reuse` on Oct 7, 2015

Conversation 69 | Commits 42 | Files changed 21

bifurcation commented on Sep 10, 2015 — Owner

This PR addresses the signature reuse vulnerability noted by Andrew Ayer.

https://mailarchive.ietf.org/arch/msg/acme/F71iz6qq1o_QPVhJCV4dqWf-4Yc

It removes the unnecessary signatures on validation objects used in challenges, and replaces it with a simpler structure where the domain holder specifies which account keys are authorized, through which challenges.

Currently WIP, for discussion.

first vuln reported in production

letsencrypt / **boulder**

Watch 94 | Star 1,034 | Fork 129

Code | Issues 258 | Pull requests 11 | Wiki | Pulse | Graphs

CAA records not verified #1231

New Issue

Closed **AlexanderS** opened this issue on Dec 7, 2015 · 4 comments

AlexanderS commented on Dec 7, 2015

I just checked if letsencrypt.org verifies CAA Records and it seems that it does **not** respect it. I just got a certificate for a domain that has a CAA record, that does **not** allow letsencrypt.org to issue a certificate.

The requested domain is *asulfrian.userpage.fu-berlin.de* and this are the CAA records for fu-berlin.de:

```
fu-berlin.de.       86400   IN  CAA 0 issue "pki.dfn.de"
fu-berlin.de.       86400   IN  CAA 0 iodef "mailto:certificate@fu-berlin.de"
```

There are no CAA records for the sub domains. Reading the relevant section in rfc6844 shows clearly that this records should be the relevant ones. So I should not be able to get this certificate.

Labels

None yet

Milestone

Sprint 2015-12-02

Assignee

rolandshoemaker

Notifications

Subscribe

- Reported 9:45 PST on 12/7
- Fix deployed 13:11 PST on 12/7
- 6 certs misissued; all revoked

Things we haven’t solved…

Things we haven’t solved...

- Mixed content :(

Mixed content problems

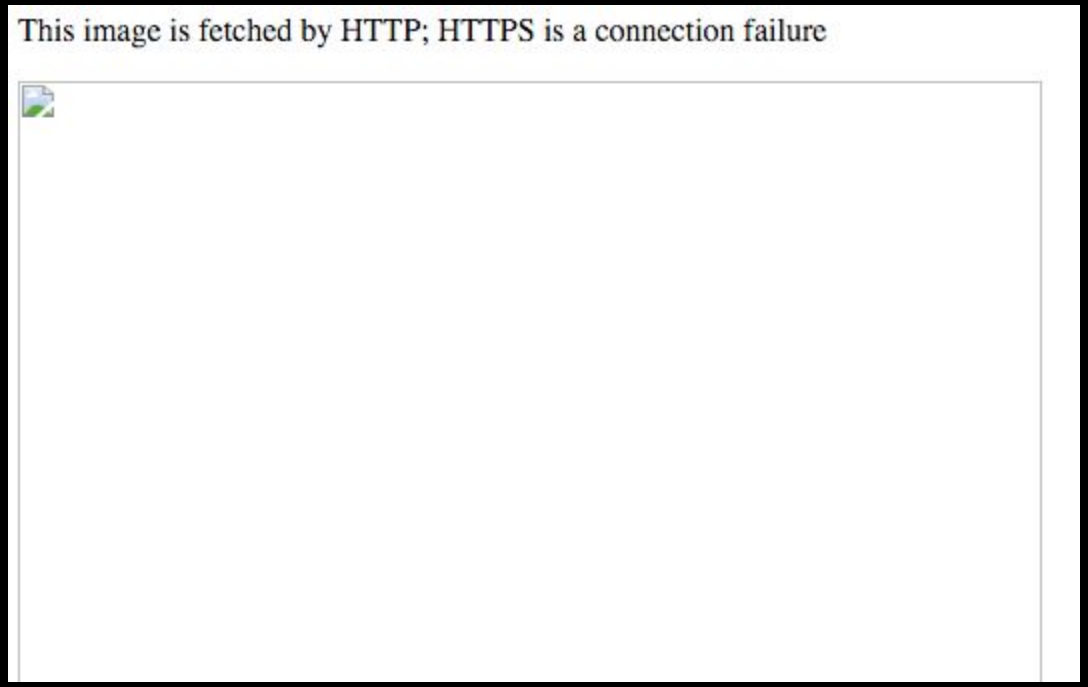
Content Security Policy `upgrade-insecure-requests`

- Was supposed to help

Except . . .

- Passive mixed content (images) isn't blocked usually
- Many (most?) HTTP embedded images aren't available over HTTPS
- Upgrade => more broken than before

https://isnot.org/mixed-uir/



Mixed Content Problems

In theory, report-only CSP is promising

In practice, auto-collecting the reports is tricky

Want to get hacking?

Spec

https://github.com/letsencrypt/acme-spec

Main Client

https://github.com/letsencrypt/letsencrypt

Server

https://github.com/letsencrypt/boulder

(And help us Encrypt the Web, entirely)