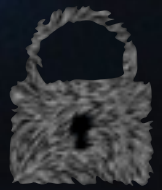
Bypassing HTTP Strict Transport Security

Jose Selvi
Twitter: @JoseSelvi

$ whois jselvi

- Jose Selvi
- +10 years working in security
- Principal Penetration Tester
- SANS Institute Community Instructor
- GIAC Security Expert (GSE)
- Twitter: @JoseSelvi
- Blog: http://www.pentester.es



Not a Silver Bullet

Let's Go!

- History of Bypassing SSL
- HTTP Strict Transport Security
- HSTS Weakness
- **********
- **********
- **********

False SSL Certificate

Client

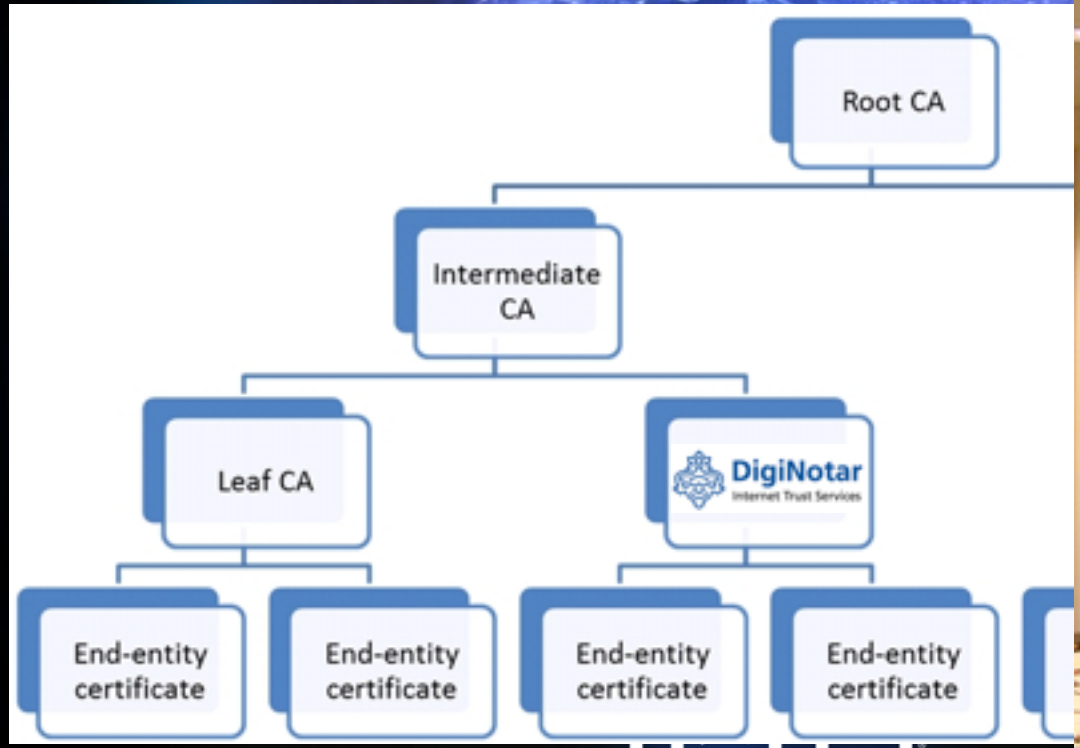
HTTPS

Attacker

HTTPS

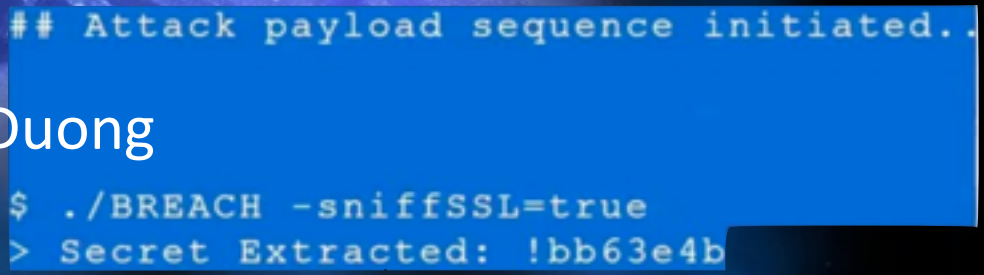
Server

PKI Compromise



Design weaknesses

- BEAST / CRIME
  - By Juliano Rizzo & Thai Duong
- BREACH
  - By Angel Prado, Neal Harris & Yoel Gluck
- Based on compression characteristics before encryption
- Chosen plaintext attack
- It can decrypt secrets (cookie, csrf-token, etc)

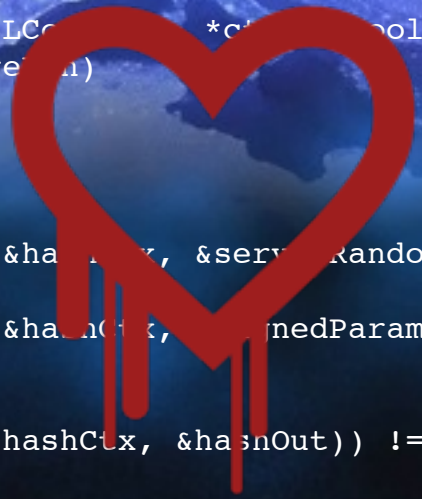


Implementation weaknesses

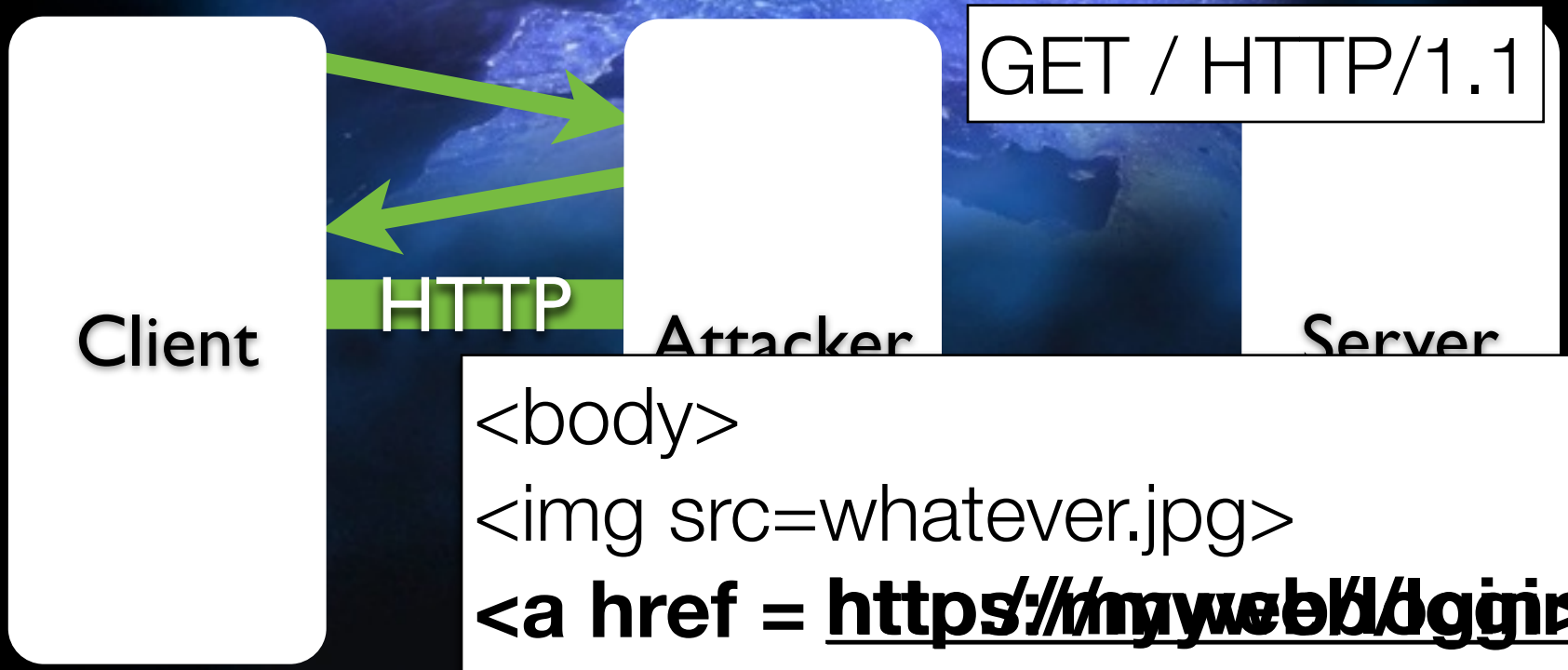
```
static OSStatus
SSLVerifySignedServerKeyExchange(SSLCo                *c      ool   isRsa,   SSLBuffer   signedParams,
uint8_t *signature, UInt16 signature                )
{
     OSStatus        err;
     ...

     if ((err = SSLHashSHA1.update(&ha      x, &serv      Random)) != 0)
          goto fail;
     if ((err = SSLHashSHA1.update(&ha      x,       nedParams)) != 0)
          goto fail;
          goto fail;
     if ((err = SSLHashSHA1.final(&hashCtx, &hashOut)) != 0)
          goto fail;
     ...

fail:
     SSLFreeBuffer(&signedHashes);
     SSLFreeBuffer(&hashCtx);
     return err;
}
```



Stripping SSL Links



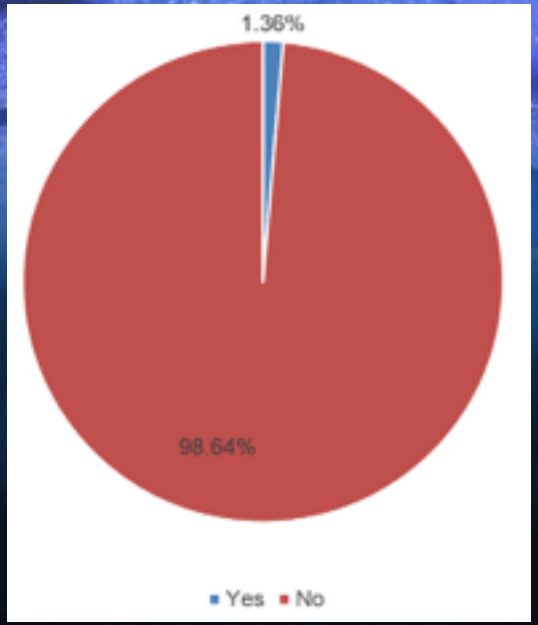
Let's Go!

- ~~History of Bypassing SSL~~
- HTTP Strict Transport Security
- HSTS Weakness
- **********
- **********
- **********

HTTP Strict Transport Security

- RFC-6797: Published in November 2012.
- Also known as HSTS or STS.
- Prevent HTTP connections.
- Prevent accepting self-signed and rogue certificates.
- Use a new “Strict-Transport-Security” header.

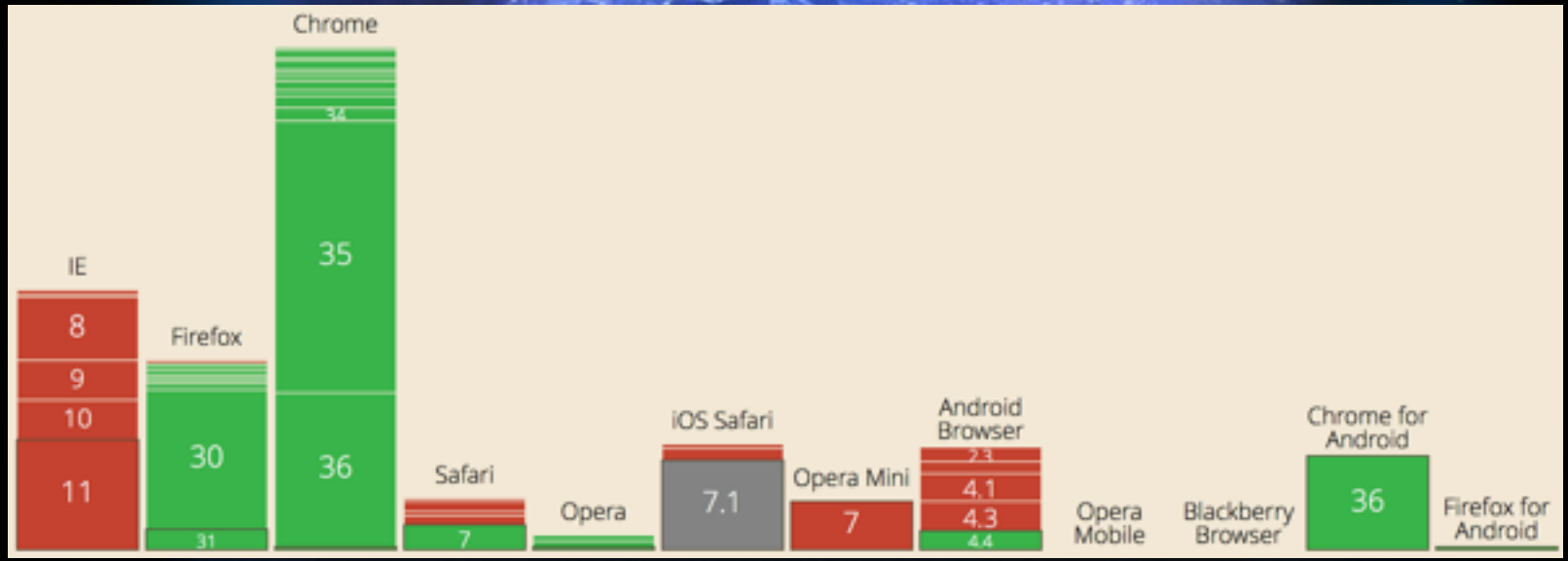
Who uses HSTS?



http://paul.vanbrouwershaven.com/2014/05/everyone-needs-http-strict-transport.html

Who uses HSTS?

Browsers support



http://caniuse.com/#feat=stricttransportsecurity

HTTPS Strict Transport Security

HTTP Strict Transport Security

- **max-age**: number of seconds that the policy is enabled.

  max-age=0 -> Delete policy

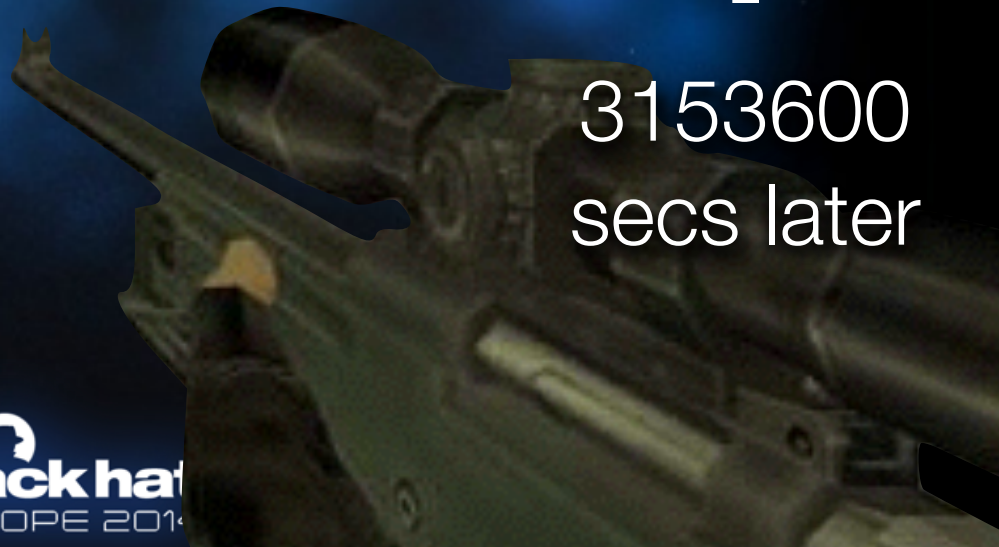
- **includeSubdomains**: If present, the policy applies all subdomains, not just the visited one.

```
$ ./hsts_catcher.py -U https://accounts.google.com
max-age=10893354; includeSubDomains
$
$ ./hsts_catcher.py -U https://paypal.com
max-age=14400
$
$ ./hsts_catcher.py -U https://github.com
max-age=31536000; includeSubdomains; preload
```

HSTS Timeline

HTTPS
connection

3153600
secs later



Preloaded HSTS

- Harcoded list of well known website names that should use always HTTPS.
- Prevent the security gap before the first HTTPS connection.
- Google, Twitter, Paypal, …

Let's Go!

- ~~History of Bypassing SSL~~
- ~~HTTP Strict Transport Security~~
- HSTS Weakness
- **********
- **********
- **********

Too short max-age

Looking for weaknesses


HST
line
3153600
secs later

Preloaded HSTS - Google

There is still a window where a user who has a fresh install, or who wipes out their local state, is vulnerable. Because of that, Chrome and Firefox share a "Preloaded HSTS" list. These domains will be configured for HSTS out of the box.

If you own a site that you would like to see included in the preloaded HSTS list you can submit it at https://hstspreload.appspot.com.

A selected subset of the members of the preloaded HSTS list:

- Google
- Paypal
- Twitter
- Simple
- Linode
- Stripe
- Lastpass

Check the source for the full list.

http://www.chromium.org/sts

Preloaded HSTS - Firefox

However, when connecting to an HSTS host for the first time, the browser won't know whether or not to use a secure connection, because it has never received an HSTS header from that host. Consequently, an active network attacker could prevent the browser from ever connecting securely (and even worse, the user may never realize something is amiss). To mitigate this attack, we have added to Firefox a list of hosts that want HSTS enforced by default. When a user connects to one of these hosts for the first time, the browser will know that it must use a secure connection. If a network attacker prevents secure connections to the server, the browser will not attempt to connect over an insecure protocol, thus maintaining the user's security.

https://blog.mozilla.org/security/2012/11/01/preloading-hsts/

Chromium Source Code

net_internals_ui.cc

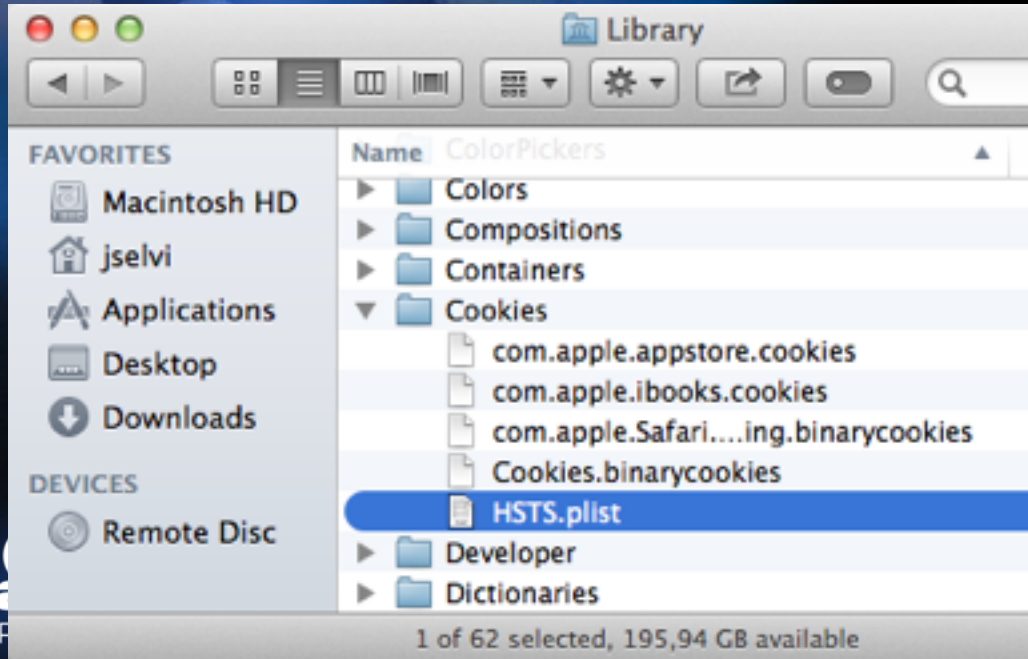
Layers ▾ Find ▾

```
1239 void NetInternalsMessageHandler::IOThreadImpl::OnHSTSAdd(
1240     const base::ListValue* list) {
1241   // |list| should be: [<domain to query>, <STS include subdomains>, <PKP
1242   // include subdomains>, <key pins>]
1243   std:
1244   CHEC
1245   if (
1246     //
1247     //
1248     re
1249   }
1250   bool
1251   CHEC
1252   bool
1253   CHEC
1254   std:
1255   CHEC
```

```
1257   net::TransportSecurityState* transport_security_state =
1258       GetMainContext()->transport_security_state();
1259   if (!transport_security_state)
1260     return;
1261
1262   base::Time expiry = base::Time::Now() + base::TimeDelta::FromDays(1000);
1263   net::HashValueVector hashes;
1264   if (!hashes_str.empty()) {
1265     if (!Base64StringToHashes(hashes_str, &hashes))
1266       return;
1267   }
1268
1269   transport_security_state->AddHSTS(domain, expiry, sts_include_subdomains);
1270   transport_security_state->AddHPKP(domain, expiry, pkp_include_subdomains,
```

Safari PList

```
$ plutil -p HSTS.plist
{
  "com.apple.CFNetwork.defaultStorageSession" => {
    "ssl.google-analytics.com" => -inf
    "webmail.mayfirst.org" => -inf
    "braintreegateway.com" => -inf
    "code.google.com" => -inf
    "dm.mylookout.com" => inf
    "therapynotes.com" => inf
    "chrome.google.com" => -inf
    "sol.io" => -inf
    "www.sandbox.mydigipass.com" => inf
[…]
```




black hat
EUROPE 2014
DEMO

HSTS Weakness

- Its security relies on time.
- It completely trust the OS's current time.
- Is it trustable?
- Is it possible to change the system time from the network?

Let's Go!

- ~~History of Bypassing SSL~~
- ~~HTTP Strict Transport Security~~
- ~~HSTS Weakness~~
- Network Time Protocol (NTP)
- Get in a Delorean
- OS Time Synchronisation & Browsers

Network Time Protocol (NTP)

- Time Synchronisation Services.
- RFC-1305 (v3) / RFC-5905 (v4) / RFC-4330 (SNTPv4).
- Set up by default on most (or all) Operating Systems.
- Security features (v4) NOT used by default.
- Vulnerable to Man-in-the-Middle techniques.

Network Time Protocol (NTP)

NTP Packet (I)

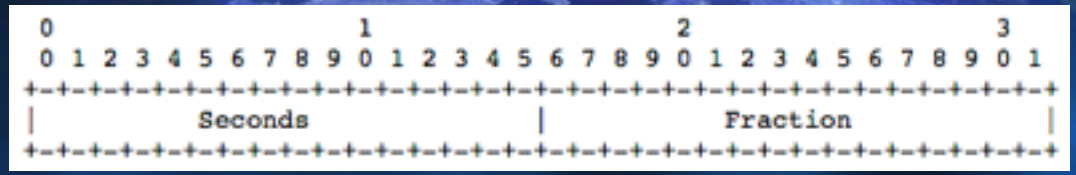
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
LI		VN			Mode			Stratum								Poll								Precisión							
Root Delay																															
Root Dispersion																															
Reference Identifier																															
Reference Timestamp (64)																															
Originate Timestamp (64)																															
Receive Timestamp (64)																															
Transmit Timestamp (64)																															
Key Identifier (optional) (32)																															
Message Digest (optional) (128)																															

NTP Packet (II)

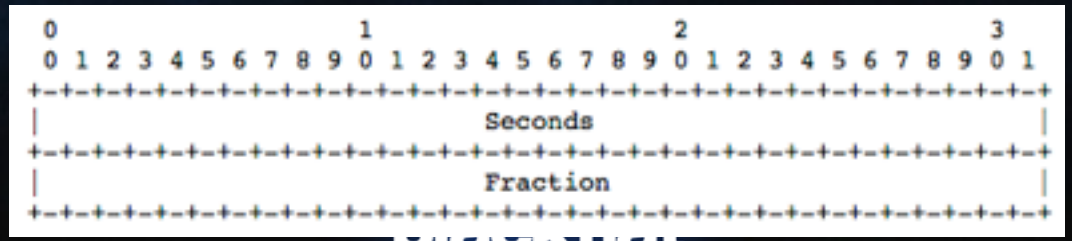
- Leap: 3 -> Clock not synchronised.
- Version: 3 /4
- Mode: Client (3) / Server (4) / etc.
- Stratum: Usually 2-15.
- Poll: NTP polling interval.
- Precision: Usually -20 (2^{-20}) -> Microseconds.

NTP Packet (III)

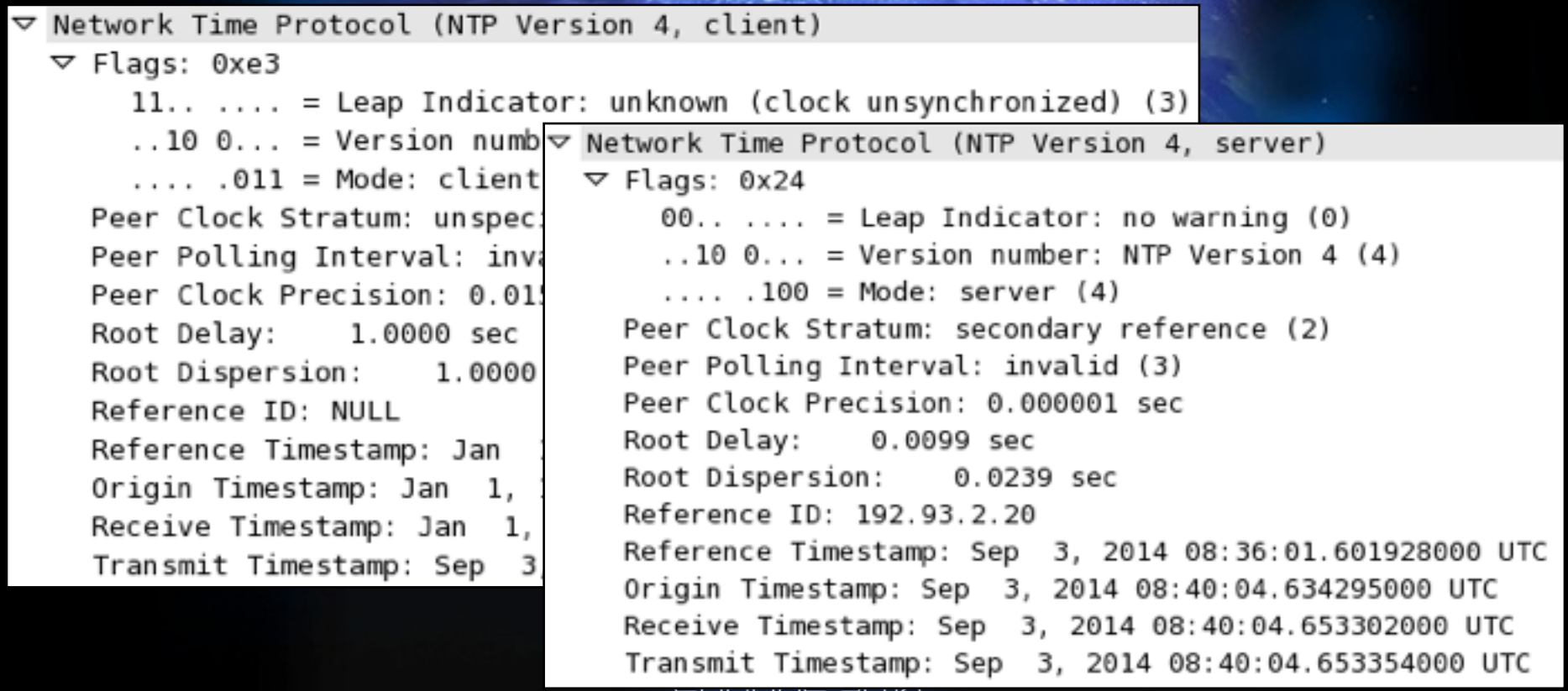
- Root delay & dispersion: NTP short format.



- Reference ID: 4 octets IPv4.
- Timestampts: NTP timestamp format



Example: Ubuntu Linux



NTP Man-in-the-Middle

Oct 21 2015 07:28

VICTIM

It's 11:00

NTP

FAKE NTP

Actually It's
Oct 21 2015 07:28

Let's Go!

- ~~History of Bypassing SSL~~
- ~~HTTP Strict Transport Security~~
- ~~HSTS Weakness~~
- ~~Network Time Protocol (NTP)~~
- Get in a Delorean
- OS Time Synchronisation & Browsers

Delorean

- NTP MitM Tool. Free. Open Source. Python.
  - http://github.com/PentesterES/Delorean
- Inspired on a kimifly's work:
  - http://github.com/limifly/ntpserver
- Implements some attacks.
- Pretend to become an NTP attack suite.

Delorean

```
$ ./delorean.py -h
Usage: delorean.py [options]

Options:
  -h, --help            show this help message and exit
  -i INTERFACE, --interface=INTERFACE
                        Listening interface
  -p PORT, --port=PORT  Listening port
  -n, --nobanner        Not show Delorean banner
  -s STEP, --force-step=STEP
                        Force the time step: 3m (minutes), 4d (days), 1M
                        (month)
  -d DATE, --force-date=DATE
                        Force the date: YYYY-MM-DD hh:mm[:ss]
  -r, --random-date     Use random date each time
```

black hat
EUROPE 2014

DEMO

Let's Go!

- ~~History of Bypassing SSL~~
- ~~HTTP Strict Transport Security~~
- ~~HSTS Weakness~~
- ~~Network Time Protocol (NTP)~~
- ~~Get in a Delorean~~
- OS Time Synchronisation & Browsers

Ubuntu Linux

- Really simple.
- NTPv4. No authentication.
- At each network reconnection (& boot time).


```
$ ls /etc/network/if-up.d/
000resolvconf  avahi-daemon  ntpdate  wpasupplicant
avahi-autoipd  ethtool       upstart
```

Fedora Linux

- The simplest one.
- NTPv3.
- More than one NTP server.
- EACH minute!


```
$ tcpdump -i eth0 -nn src port 123
12:43:50.614191 IP 192.168.1.101.123 > 89.248.106.98.123: NTPv3, Client, length 48
12:44:55.696390 IP 192.168.1.101.123 > 213.194.159.3.123: NTPv3, Client, length 48
12:45:59.034059 IP 192.168.1.101.123 > 89.248.106.98.123: NTPv3, Client, length 48
```

Mac OS X - Lion

- Pretty simple as well.
- NTPv4. No authentication.
- Each 9 minutes.

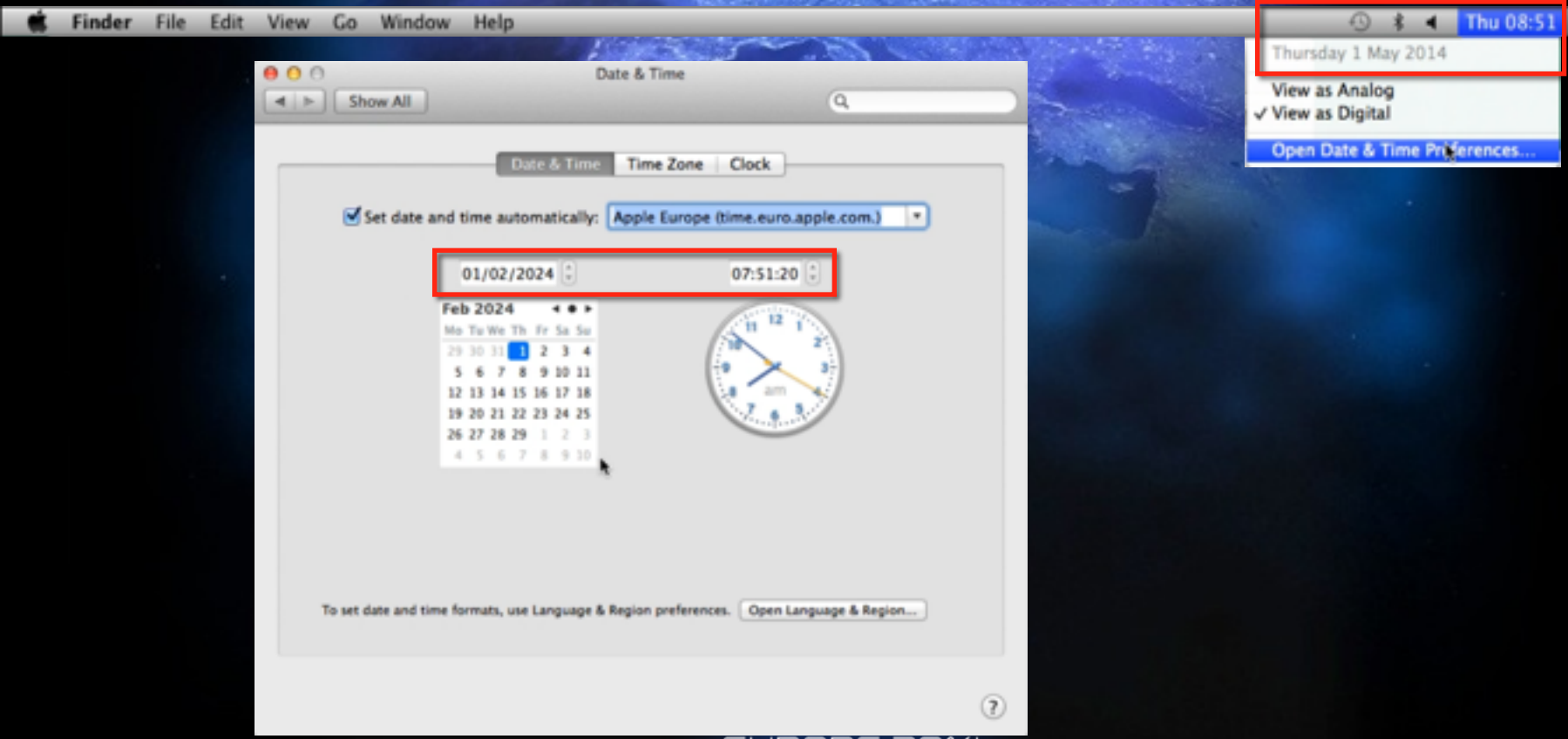
```
$ tcpdump -i eth0 -nn src port 123
09:02:18.166708 IP 192.168.1.100.123 > 17.72.148.53.123: NTPv4, Client, length 48
09:11:20.059792 IP 192.168.1.100.123 > 17.72.148.53.123: NTPv4, Client, length 48
09:20:17.951361 IP 192.168.1.100.123 > 17.72.148.53.123: NTPv4, Client, length 48
```

Mac OS X - Mavericks

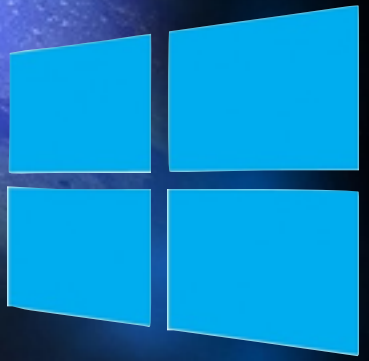
- New synchronisation service.
- NTP still exists but not synchronises.
  - Just write in /var/db/ntp.drift
- A new service called "**pacemaker**" check this file and synchronise the system clock.
- It seems it doesn't work as expected…

http://www.atmythoughts.com/living-in-a-tech-family-blog/2014/2/28/what-time-is-it

Mac OS X - Mavericks

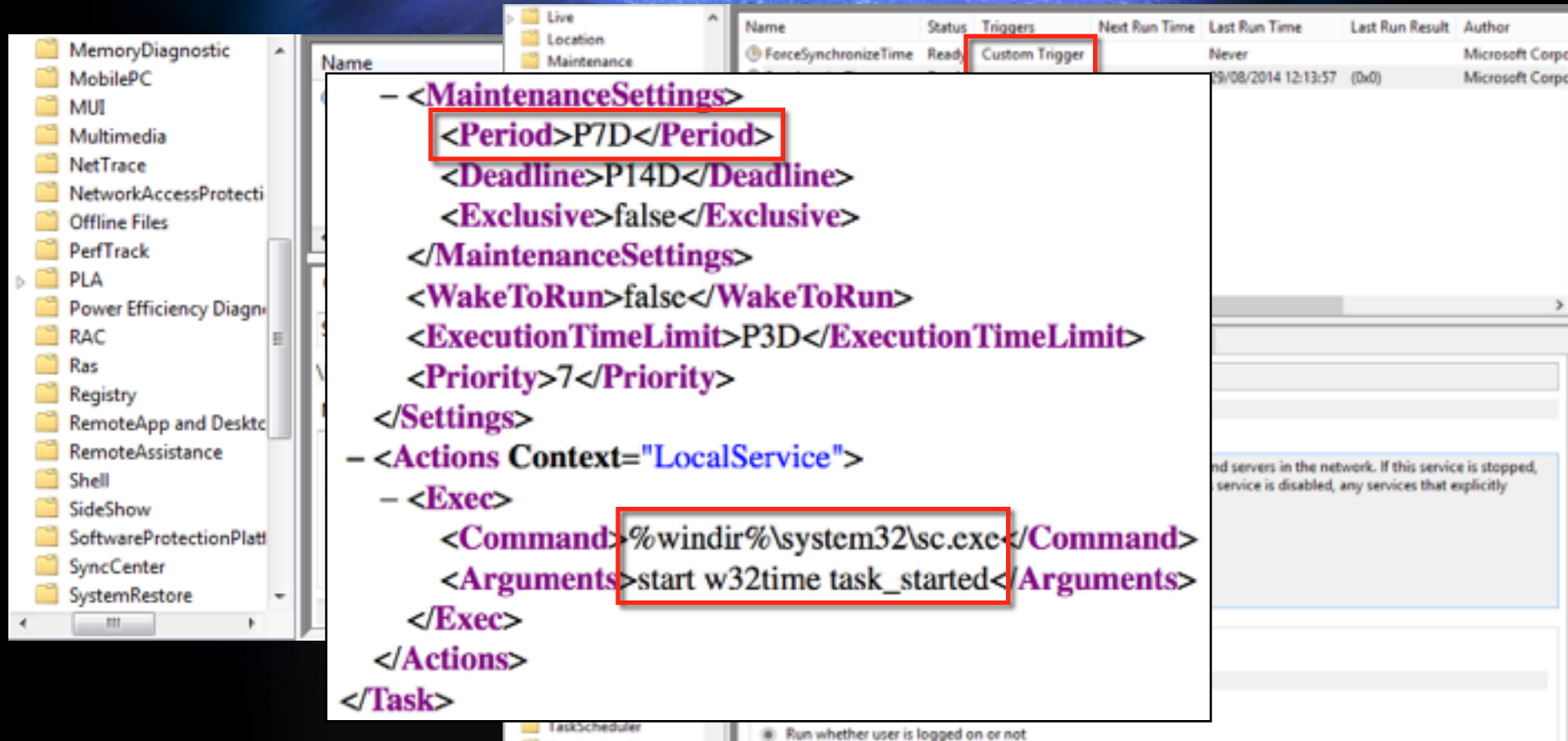


Windows

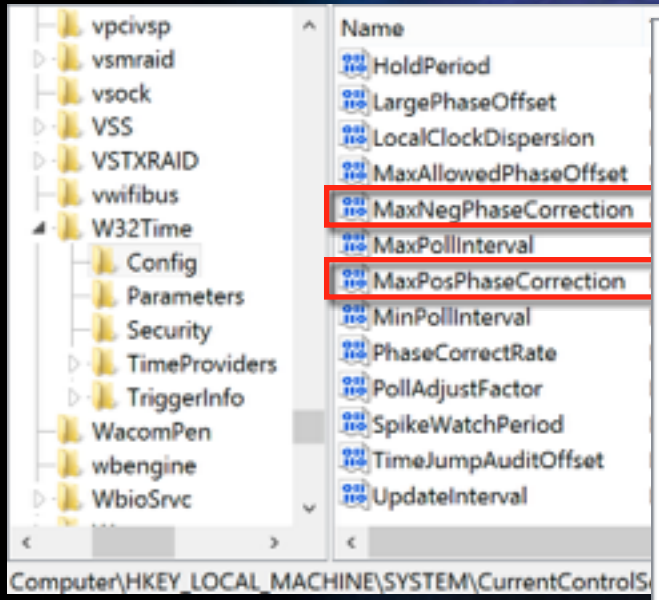


- NTPv3 but…
- The securest one.
- Synchronization each 7 days.
- Doesn’t accept more than 15 hours increment/ decrement.
- Domain members have a different set up.

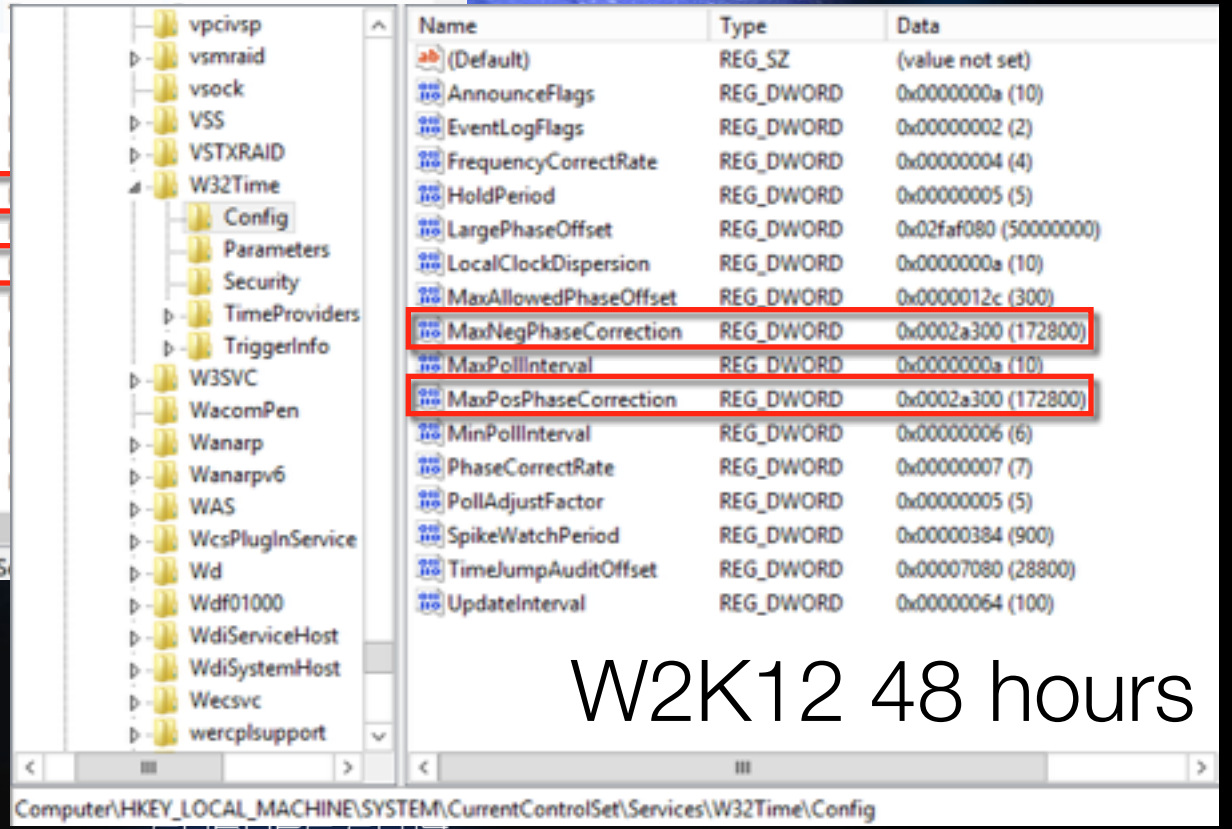
W32time Service



Max[Pos|Neg]PhaseCorrection



W7 / W8
15 hours



W2K12 48 hours

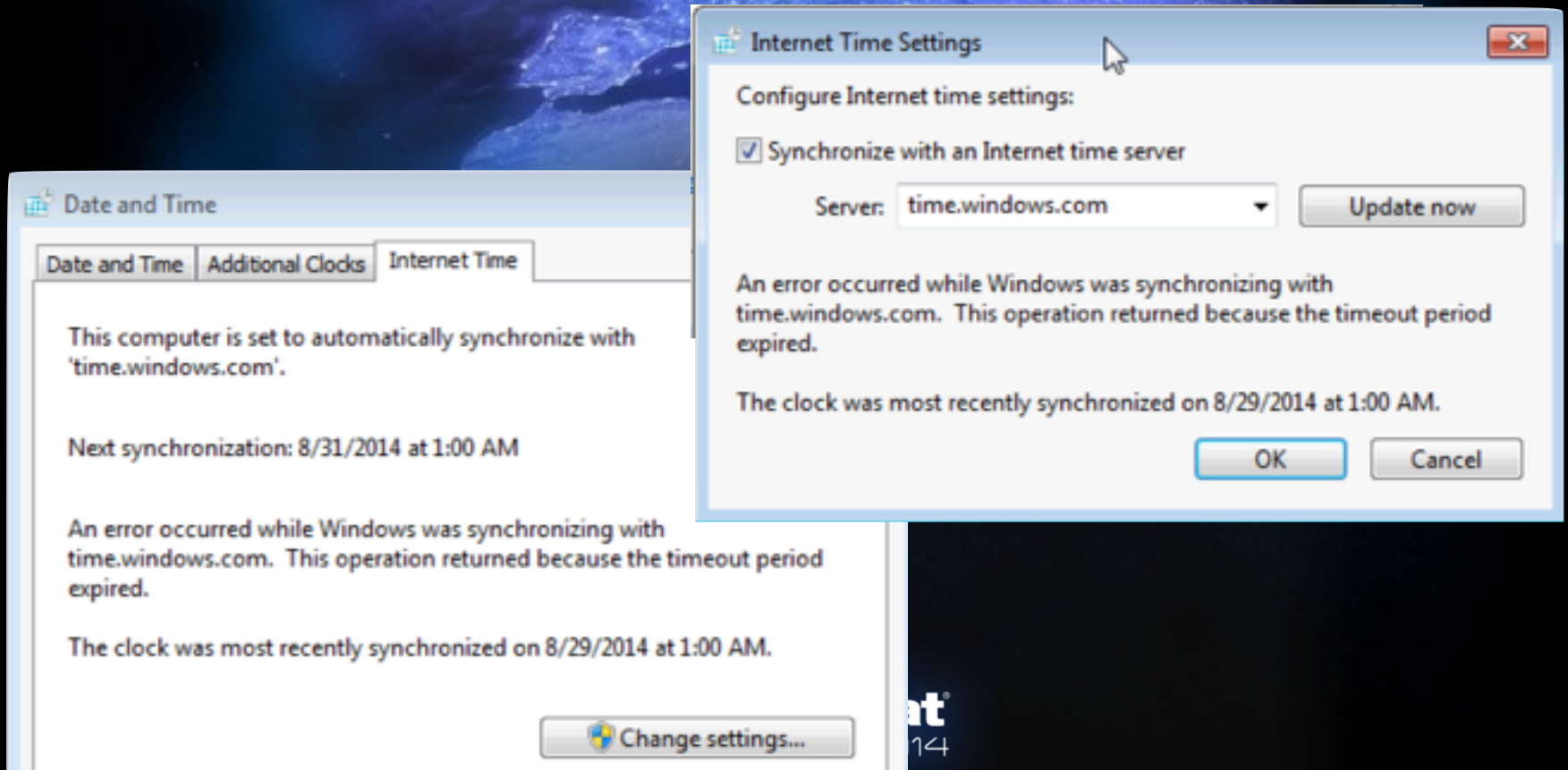
Time Skimming Attack

Time Syr

153600

ecs later

Force Synchronisation




black hat
EUROPE 2014
DEMO

Let's Go!

- ~~History of Bypassing SSL~~
- ~~HTTP Strict Transport Security~~
- ~~HSTS Weakness~~
- ~~Network Time Protocol (NTP)~~
- ~~Get in a Delorean~~
- ~~OS Time Synchronisation & Browsers~~

Whose fault is?

Answer from Google

#1 @chromium.org — Yesterday (17 hours ago)

This is a pretty cool DoS.

@ do you want to look at this?

I imagine we need some pre-established channel to establish the time. I am guessing this might be available already through Omaha or similar but I'm not sure.

Status: Available
Cc: @chromium.org @gmail.com
Labels: Security_Impact-Stable Security_Impact-Beta Security_Severity-Low

#2 jse...@gmail.com — Today (16 hours ago) Delete comment

In fact, If you force hsts entries to expire, you can use ssl-stripping attacks and capture credentials and other information. DoS is possible as well, but I think the credential capture is the most critical attack surface.

#3 @chromium.org — Today (19 minutes ago)

Thanks for the report, but this is a fairly well known, generic issue with relying on unauthenticated NTP, and is not specific to HSTS. Consider that SSL/TLS, Kerberos, and MS ActiveDirectory all have the same underlying concerns and attack profile. The typical solutions for this are to use authenticated NTP, use multiple NTP servers with a client that detects excessive drift or "bad tickers", or to use some other mechanism to provide authenticated time.

Status: WontFix

References

- https://www.owasp.org/index.php/HTTP_Strict_Transport_Security
- https://tools.ietf.org/html/rfc6797
- http://dev.chromium.org/sts
- https://developer.mozilla.org/en-US/docs/Web/Security/HTTP_strict_transport_security
- http://www.ntp.org
- https://github.com/limifly/ntpserver
- http://www.thoughtcrime.org/software/sslstrip/
- https://github.com/LeonardoNve/dns2proxy

Thanks! Questions?

Jose Selvi
http://twitter.com/JoseSelvi

jselvi@incide.es
http://www.incide.es

jselvi@pentester.es
http://www.pentester.es