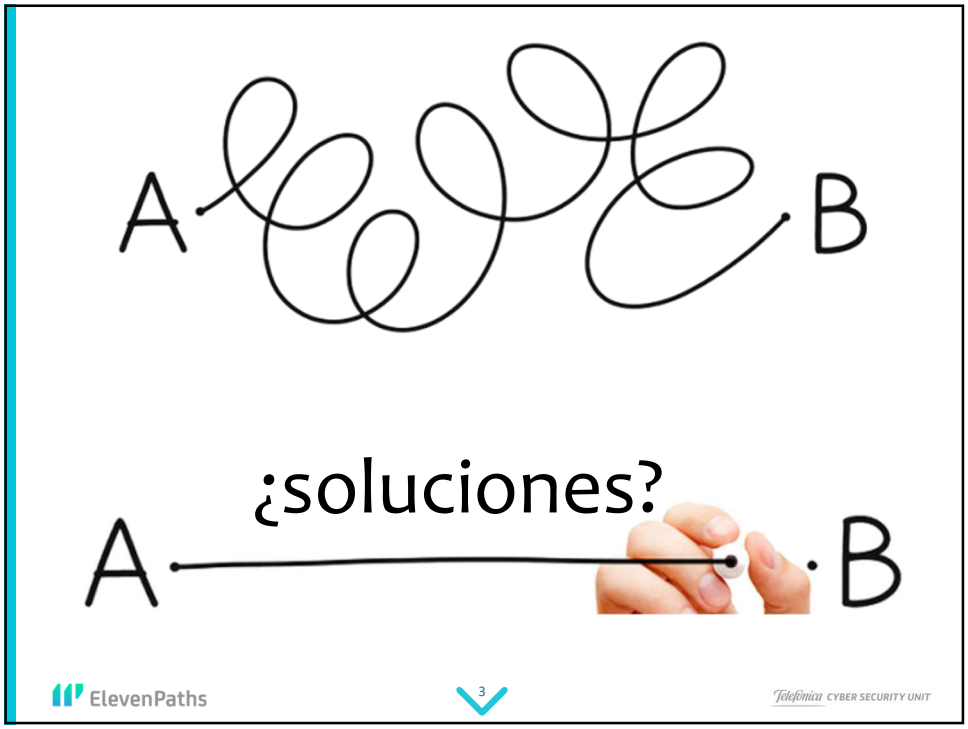
Lo que todos ya sabemos

- ¿Para qué sirve HTTPS?
- ¿Cómo funciona la negociación SSL/TLS?
- ¿Qué ataques existen para SSL/TLS?
- ¿Qué es HSTS?
- ¿Qué es Certificate Pinning?
- **Y la más importante, ¿Qué hacen las CA?**



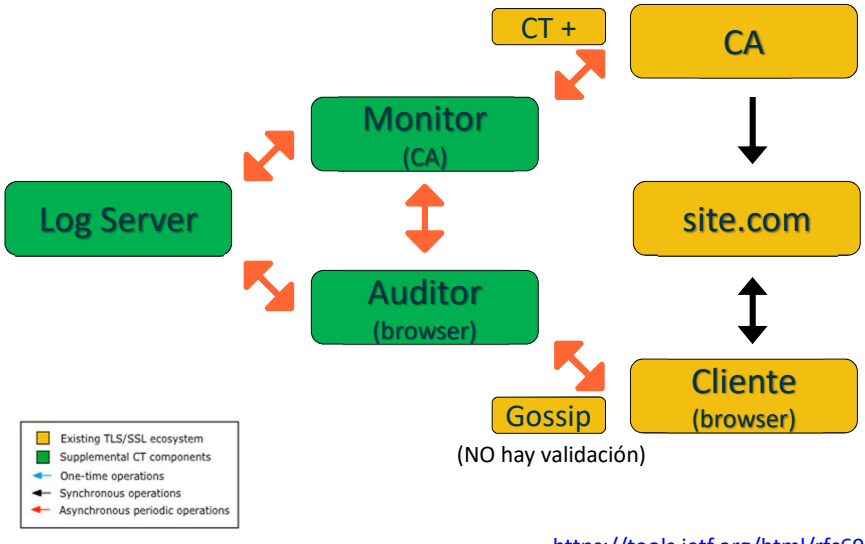
Certificate Transparency (RFC 6962)

- CT es un protocolo (*gossip*) para registrar públicamente la existencia de certificados TLS a medida que se emiten, de manera que cualquiera pueda auditar la actividad de la CA y notificar la emisión de certificados sospechosos
- La intención es que los navegadores no puedan validar certificados que no aparecen en el registro. Actualmente funciona sólo con EV
- Impulsado por Google desde la versión 45 (09/15) y de obligatorio desde octubre de 2017

https://tools.ietf.org/html/rfc6962

ElevenPaths 4 Telefónica CYBER SECURITY UNIT

Funcionamiento de CT

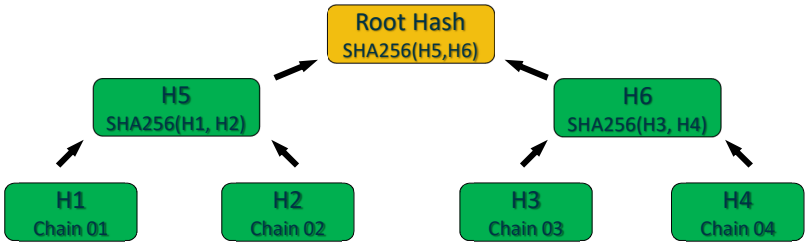


https://tools.ietf.org/html/rfc6962

ElevenPaths 5 Telefónica CYBER SECURITY UNIT

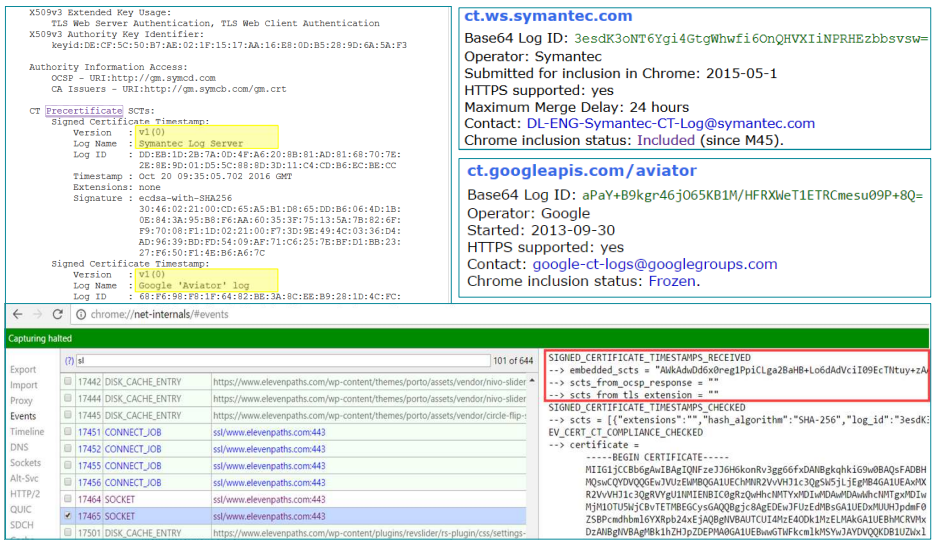
Árbol Merkle

- Los *logs* se almacenan en una estructura de árbol Merkle binario, único y creciente, similar a lo que sucede con el *Blockchain*
- Las hojas del árbol corresponden a los *hashes* SHA-256 de los certificados que han sido agregados al *log*



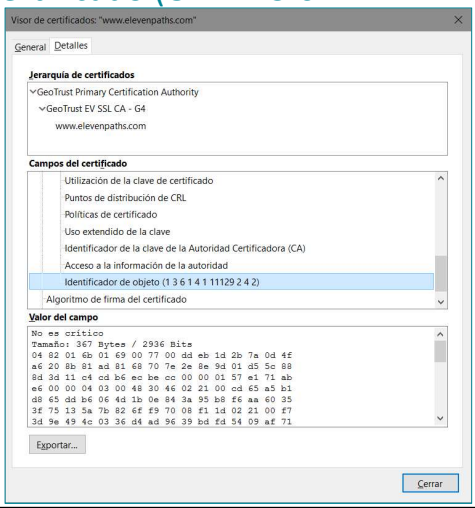
ElevenPaths 6 Telefónica CYBER SECURITY UNIT

Registro SCT *(Signed Certificate Timestamp)*



Lugar del SCT

- **Como una Extensión X.509v3:** la CA lo “incrusta” al crear el certificado (OID: 1.3.6.1.4.1.11129.2.4.2)



Lugar del SCT

- **A través de la extensión 0x12 del protocolo TLS:** la CA entrga un certificado normal y el administrador del servidor debe enviarlo al *log* de forma manual
- **En la respuesta OCSP:** la CA envía el certificado al sitio y al *log* y luego el servidor realiza una petición OCSP a la CA

ElevenPaths 9 Telefónica CYBER SECURITY UNIT

ElevenPaths Telefónica CYBER SECURITY UNIT

El código fuente

https://chromium.googlesource.com/chromium/src/+/master/net/cert/ct_known_logs_static-inc.h

```
// The set of all presently-qualifying CT logs.
// Google provides DNS frontends for all of the logs.
const CTLogInfo kCTLogList[] = {
    {"\x30\x59\x30\x13\x06\x07\x2a\x86\x48\xce\x3d\x02\x01\x06\x08\x2a\x86"
     "\x48\xce\x3d\x03\x01\x07\x03\x42\x00\x04\x7d\xa8\x4b\x12\x29\x80\xa3"
     "\x3d\xad\xd3\x5a\x77\xb8\xcc\xe2\x88\xb3\xa5\xfd\xf1\xd3\x0c\xcd\x18"
     "\x0c\xe8\x41\x46\xe8\x81\x01\x1b\x15\xe1\x4b\xf1\x1b\x62\xdd\x36\x0a"
     "\x08\x18\xba\xed\x0b\x35\x84\xd0\x9e\x40\x3c\x2d\x9e\x9b\x82\x65\xbd"
     "\x1f\x04\x10\x41\x4c\xa0",
     91, "Google 'Pilot' log", "https://ct.googleapis.com/pilot/",
     "pilot.ct.googleapis.com"},
    {"\x30\x59\x30\x13\x06\x07\x2a\x86\x48\xce\x3d\x02\x01\x06\x08\x2a\x86"
     "\x48\xce\x3d\x03\x01\x07\x03\x42\x00\x04\xd7\xf4\xcc\x69\xb2\xe4\x0e"
     "\x90\xa3\x8a\xea\x5a\x70\x09\x4f\xef\x13\x62\xd0\x8d\x49\x60\xff\x1b"
     "\x40\x50\x07\x0c\x6d\x71\x86\xda\x25\x49\x8d\x65\xe1\x08\x0d\x47\x34"
     "\x6b\xbd\x27\xbc\x96\x21\x3e\x34\xf5\x87\x76\x31\xb1\x7f\x1d\xc9\x85"
     "\x3b\x0d\xf7\x1f\x3f\xe9",
     91, "Google 'Aviator' log", "https://ct.googleapis.com/aviator/",
     "aviator.ct.googleapis.com"},
```

ElevenPaths 11 Telefónica CYBER SECURITY UNIT

La cantidad de días de verificación

shopping [redacted].net
whois information

Whois | Website Info | History | DNS Records | Diagnostics

cache expires in 19 hours, 16 minutes and 52 seconds

Registrar Info

Name	Ascio Technologies, Inc
Whois Server	whois.ascio.com
Referral URL	http://www.ascio.com
Status	clientTransferProhibited http://www.icann.org/epp

Important Dates

```
openssl s_client -connect shopping [redacted].net:443
notBefore=Apr  9 00:00:00 2014 GMT
notAfter=Apr  8 23:59:59 2017 GMT
```

Expires On	2018-03-17
Registered On	2017-03-17
Updated On	2017-03-17

Hoy es 27/03
(pasaron 10 días)

ElevenPaths 12 Telefónica CYBER SECURITY UNIT

La cantidad de días de verificación

ct-lta.tk

crt.sh ID	109353487
Summary	Leaf certificate
Certificate Transparency	

Timestamp	Entry #	Log Operator	Log URL
2017-03-27 15:21:30 GMT	81642411	Google	https://ct.googleapis.com/rocketeer
2017-03-27 15:21:30 GMT	848036	WoSign	https://ctlog.wosign.com
2017-03-27 15:21:31 GMT	787477	Google	https://ct.googleapis.com/skydiver
2017-03-29 17:07:46 GMT	84651209	Google	https://ct.googleapis.com/pilot

crt.sh ID	112[redacted]
Summary	Leaf certificate
Certificate Transparency	

Timestamp	Entry #	Log Operator	Log URL
2017-04-04 07:35:03 GMT	41220789	Google	https://ct.googleapis.com/icarus
2017-04-06 22:45:09 GMT	235200	Venafi	https://ctlog-gen2.api.venafi.com

CT **no** se usará para revocar certificados, sólo se usará para detectar el uso inapropiado de los mismos!

¿Esto va en serio?

Add support for `Expect-CT` header #224

Open craigfrancis opened this issue 22 days ago · 7 comments

craigfrancis commented 22 days ago Contributor

Might be a little early yet, but the `Expect-CT` header is likely to be implemented in Chrome and Firefox:

https://www.chromestatus.com/feature/5677171733430272

```
Expect-CT: enforce;
           max-age=1440;
           report-URI=https://blah.com/CTReport/
```

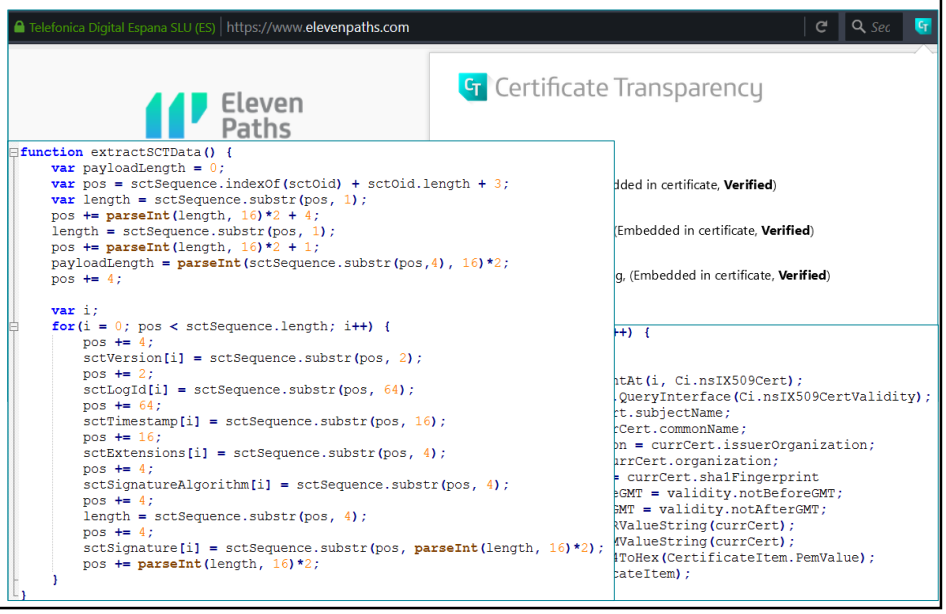
enforce: el navegador espera un SCT válido o aborta la conexión

max-age: tiempo de caché de esta directiva en el navegador (en seg.)

report-URI: URL a la que el navegador deber enviar reportes de fallos

https://tools.ietf.org/id/draft-stark-expect-ct-01.txt

Nuestro aporte



Y...

En el paper hay una **sorpresa** ☺

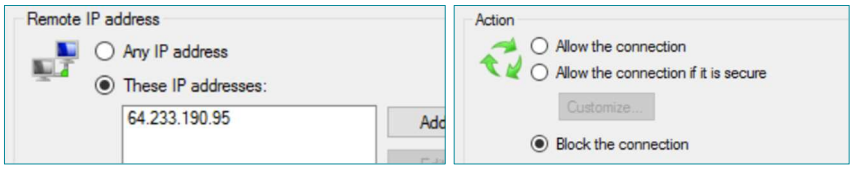
Una aproximación práctica al Certificate Transparency

https://bit.ly/certificate-transparency

ElevenPaths 17 Telefónica CYBER SECURITY UNIT

Implementación en Chrome (cont.)

ct.googleapis.com = 64.233.186.95

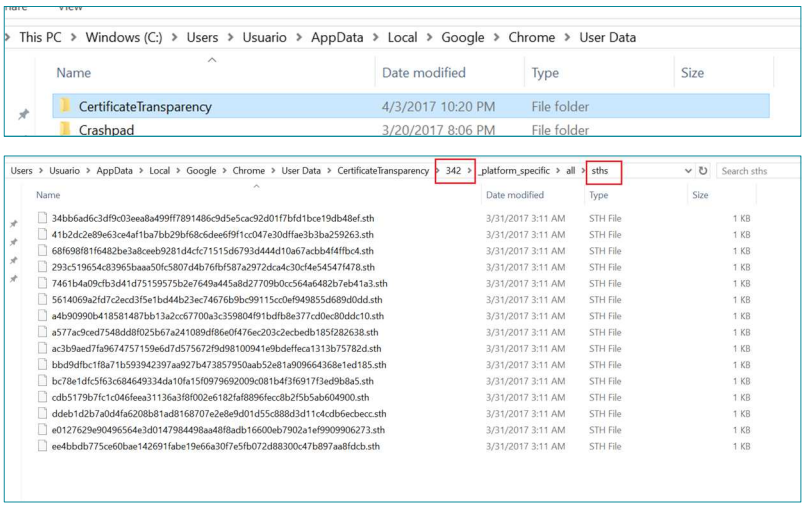


No se bloquea el funcionamiento, ya que la verificación **no** se realiza a través de esta conexión

Google verifica de forma asíncrona para agregar más sitios a sus servidores de Transparencia

ElevenPaths 18 Telefónica CYBER SECURITY UNIT

Implementación en Chrome (cont.)

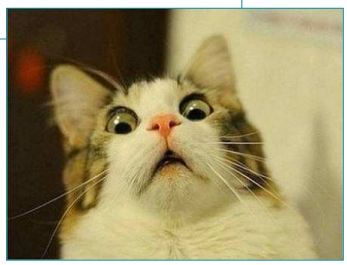


Implementación en Chrome (cont.)

File: sth_set_component_installer.h

```
67   // Reads and parses the on-disk json.
68   void LoadSTHsFromDisk(const base::FilePath& sths_file_path,
69                         const base::Version& version);
70
```

```
115
116 void STHSetComponentInstallerTraits::LoadSTHsFromDisk(
117     const base::FilePath& sths_path,
118     const base::Version& version) {
119   if (sths_path.empty())
120     return;
```



Implementación en Chrome (cont.)

```
121
122  base::FileEnumerator sth_file_enumerator(sths_path, false,
123                                           base::FileEnumerator::FILES,
124                                           FILE_PATH_LITERAL("*.sth"));
125  base::FilePath sth_file_path;
```

```
126
127  while (!(sth_file_path = sth_file_enumerator.Next()).empty()) {
128    DVLOG(1) << "Reading STH from file: " << sth_file_path.value();
129
130    const std::string log_id_hex =
131        sth_file_path.BaseName().RemoveExtension().MaybeAsASCII();
132    if (log_id_hex.empty()) {
133      DVLOG(1) << "Error extracting log_id from: "
134               << sth_file_path.BaseName().LossyDisplayName();
135      continue;
136    }
137
138    std::vector<uint8_t> decoding_output;
139    if (!base::HexStringToBytes(log_id_hex, &decoding_output)) {
140      DVLOG(1) << "Failed to decode Log ID: " << log_id_hex;
141      continue;
142    }
```


Conclusiones

- *Work in progress…* veremos en octubre
- CT debería ayudar a la detección de certificados fraudulentos en un mejor tiempo
- CT no reemplaza nada de certificados digitales, lo complementa
- Más trabajo para los monitores (CA)
- Solo Google por ahora ha implementado CT y su funcionamiento sigue siendo un poco oscuro
- CT es una fuente de información valiosa que puede ser explotada (para bien o mal)


Eleven Paths
¡GRACIAS!
Lic. Cristian Borghello, CISSP - CCSK
@crisborghe / @seguinfo
Sheila Berta
@unapibageek
Telefónica CYBER SECURITY UNIT