Private Key Leaks in the Wild

Insights from Certificate Transparency

Philippe BONEFF - **Gaetan FERRY** - David TAO - **Guillaume VALADON**

$ whoami

Guillaume

Cybersecurity Researcher

editor-in-chief of the **MISC magazine**

Scapy maintainer

previously at **Quarkslab,** ANSSI…

Gaetan

Cybersecurity Researcher

former researcher **@Sonar**

Synacktiv red teamer for 7 years

01

Private Key Leaks and Certificate Transparency

From unknown keys to identities

Leaks Everywhere You Look

Public secrets leaks an **underestimated problem**

29 millions

secrets leaked on GitHub in 2025

⬆️ 34% increase from 2024

430,000

private keys included

Private Key Content

Mathematical objects

Several use cases: TLS, GPG, SSH...

Several encoding format

DER, PEM, PKCS#1, encrypted, plaintext...

Stores cryptographic parameters

RSA, ECDSA...

```
cat key.pem
-----BEGIN PRIVATE KEY-----
MIIEvAIBADANBgkqhkiG9w0BAQEFAASCBKYwggSiAgEAAoIBAQCubmXJRjDkDVF7
AxSu1AwUSLoCda84IZYy5HtNj1ZnzPdu6nMcQ2guIW2OyJYDi+LILpqESdgS6TJQ
8< - - - 8< - - - 8< - - -
uq39bRt6zJzSWEzAjveuSBKYvcA3Umc4zfL1M7TsaHdX/V5VVEiipHcgdHpg++jq
D0nHHnlALMePBLT2eNxgYw==
-----END PRIVATE KEY-----
```

```
openssl rsa -noout -text -in key.pem
Private-Key: (2048 bit, 2 primes)
modulus:
    00:86:39:eb:40:9a:c6:49:9c:f5:ad:11:69:e0:b3:
    Cb:b4:f7:bd:de:b3:3b:64:13:3b:2d:a7:07:cd:b3:
    8< - - - 8< - - - 8< - - -
    c2:ba:6d:e4:8c:2e:c2:aa:5f:61:fb:61:d3:b0:1a:
    f1:67
publicExponent: 65537 (0x10001)
privateExponent:
    3f:38:90:bd:95:05:f6:8c:46:55:14:37:4c:2e:c4:
    ff:46:42:d1:fd:6a:ee:6f:75:63:c8:ae:4f:9c:ee:
    8< - - - 8< - - - 8< - - -
```

TLS Certificate Content

X.509 certificates format

- Identity to the public exponent
- Validity period
- CA's signature
- Usages
- ...

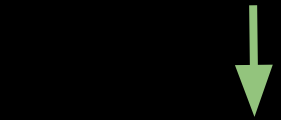
```
cat sstic.org.certificate.pem
-----BEGIN CERTIFICATE-----
MIIE+zCCA+OgAwIBAgISBgs0r6RtgIAr2vJzuf6+e5kLMA0GCSqGSIb3DQEBCwUA
MDMxCzAJBgNVBAYTAlVTMRYwFAYDVQQKEw1MZXQncyBFbmNyeXB0MQwwCgYDVQQD
8< - - - 8< - - - 8< - - -
gMQgIVcurPFJtE7AdZiLp3CwAOF5dH1FZ/QyjmtZPvJuD/q9o1mpEljXXqvbRZs9
6ier6T7zMqxE4pgAQ/+Ex/uj3rLgcSHIDSKpjX+CQQ==
-----END CERTIFICATE-----
```

```
openssl x509 -noout -text -in sstic.org.certificate.pem
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            06:0b:34:af:a4:6d:80:80:2b:da:f2:73:b9:fe:be:7b:99:0b
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C=US, O=Let's Encrypt, CN=R12
        Validity
            Not Before: Mar 21 21:35:07 2026 GMT
            Not After : Jun 19 21:35:06 2026 GMT
        Subject: CN=sstic.org
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:86:39:eb:40:9a:c6:49:9c:f5:ad:11:69:e0:b3:
                    71:9a:26:5f:21:a8:0d:f7:d8:b3:1b:cd:c8:fa:b9:
                    8< - - - 8< - - - 8< - - -
                    c2:ba:6d:e4:8c:2e:c2:aa:5f:61:fb:61:d3:b0:1a:
                    f1:67
```

Mapping a Private Key to a Certificate with OpenSSL

```
openssl rsa -modulus -noout -in leaked_private.key | openssl sha256
```

```
openssl x509 -modulus -noout -in sstic.org.pem | openssl sha256
```



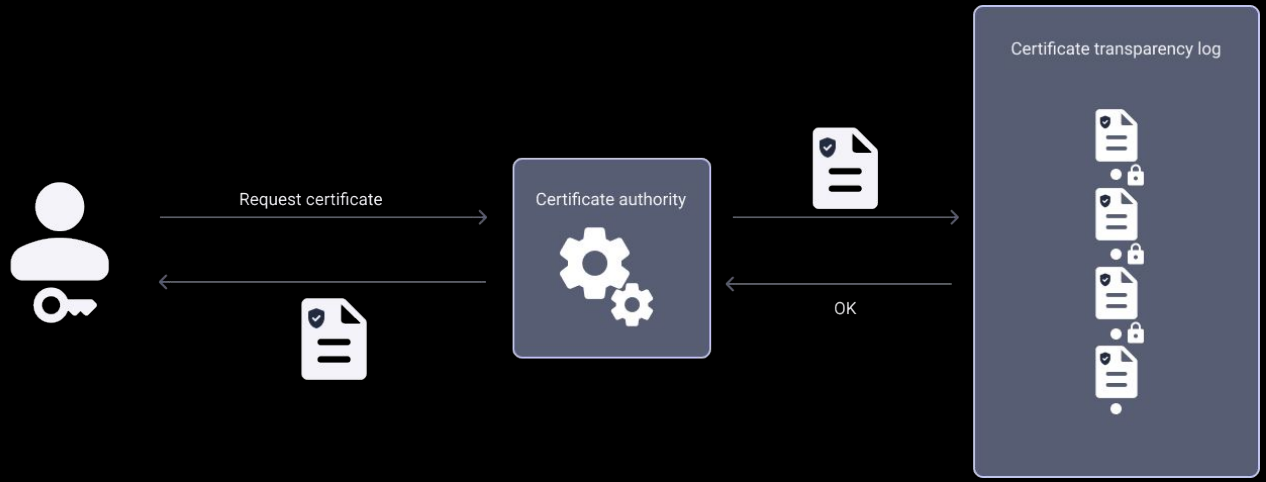
```
SHA2-256(stdin)= 9eed8ad6796bc0622984c715ef26ccd65d6019223e22802dc7f3724adda4bb26
```

🤔 Compute the SPKI hashes of the leaked private keys and...

Certificate Transparency

Core mechanism of the X.509 infrastructure since 2015

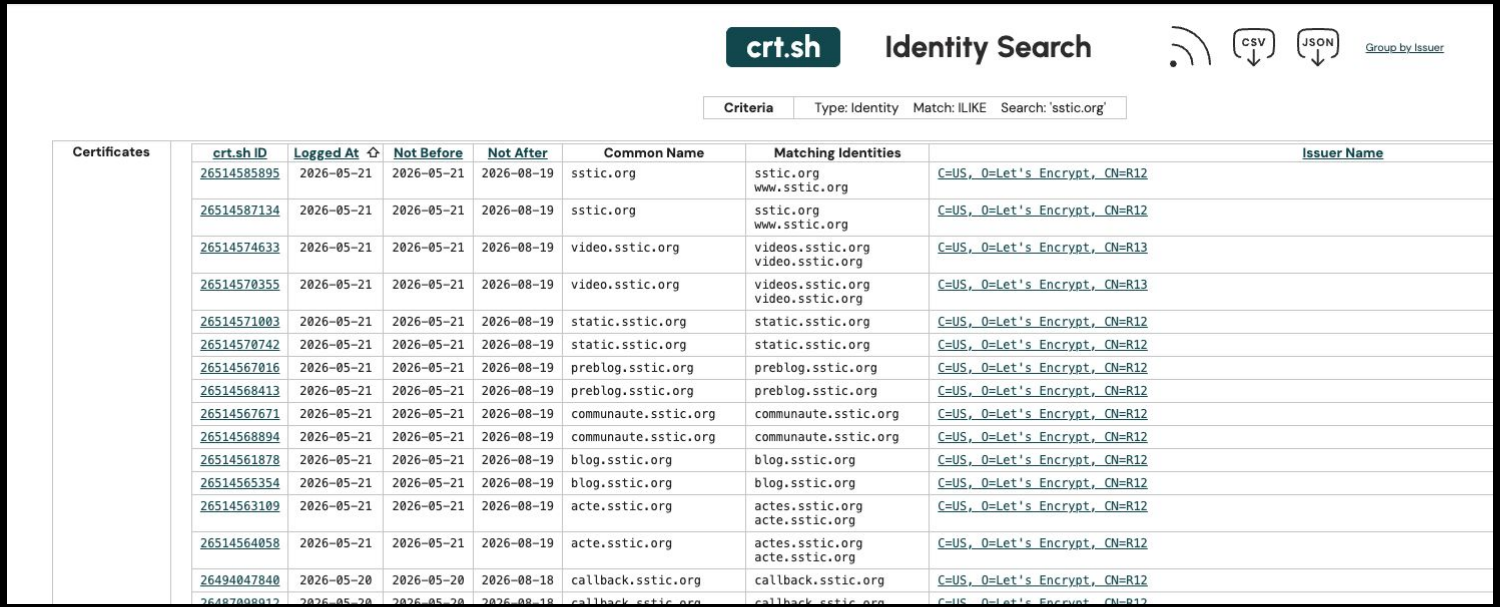
- Every CA must submit certificates it issues to a Certificate Transparency Log
- Logs are stored as Merkle-Trees: append only, tamper-proof
- Logs are operated by multiple parties. Anyone can host a log
- Browsers check a proof of inclusion when receiving certificates



Certificate Transparency

CT logs can be accessed and queried publicly

- Makes certificates issuance auditable (monitor a domain for evil certificates)
- Third party services allow searching the database (crt.sh, censys, etc)



crt.sh Identity Search

CSV JSON Group by Issuer

Criteria: Type: Identity Match: ILIKE Search: 'sstic.org'

Certificates

crt.sh ID	Logged At	Not Before	Not After	Common Name	Matching Identities	Issuer Name
26514585895	2026-05-21	2026-05-21	2026-08-19	sstic.org	sstic.org www.sstic.org	C=US, O=Let's Encrypt, CN=R12
26514587134	2026-05-21	2026-05-21	2026-08-19	sstic.org	sstic.org www.sstic.org	C=US, O=Let's Encrypt, CN=R12
26514574633	2026-05-21	2026-05-21	2026-08-19	video.sstic.org	videos.sstic.org video.sstic.org	C=US, O=Let's Encrypt, CN=R13
26514570355	2026-05-21	2026-05-21	2026-08-19	video.sstic.org	videos.sstic.org video.sstic.org	C=US, O=Let's Encrypt, CN=R13
26514571003	2026-05-21	2026-05-21	2026-08-19	static.sstic.org	static.sstic.org	C=US, O=Let's Encrypt, CN=R12
26514570742	2026-05-21	2026-05-21	2026-08-19	static.sstic.org	static.sstic.org	C=US, O=Let's Encrypt, CN=R12
26514567016	2026-05-21	2026-05-21	2026-08-19	preblog.sstic.org	preblog.sstic.org	C=US, O=Let's Encrypt, CN=R12
26514568413	2026-05-21	2026-05-21	2026-08-19	preblog.sstic.org	preblog.sstic.org	C=US, O=Let's Encrypt, CN=R12
26514567671	2026-05-21	2026-05-21	2026-08-19	communaute.sstic.org	communaute.sstic.org	C=US, O=Let's Encrypt, CN=R12
26514568894	2026-05-21	2026-05-21	2026-08-19	communaute.sstic.org	communaute.sstic.org	C=US, O=Let's Encrypt, CN=R12
26514561878	2026-05-21	2026-05-21	2026-08-19	blog.sstic.org	blog.sstic.org	C=US, O=Let's Encrypt, CN=R12
26514565354	2026-05-21	2026-05-21	2026-08-19	blog.sstic.org	blog.sstic.org	C=US, O=Let's Encrypt, CN=R12
26514563109	2026-05-21	2026-05-21	2026-08-19	acte.sstic.org	actes.sstic.org acte.sstic.org	C=US, O=Let's Encrypt, CN=R12
26514564058	2026-05-21	2026-05-21	2026-08-19	acte.sstic.org	actes.sstic.org acte.sstic.org	C=US, O=Let's Encrypt, CN=R12
26494047840	2026-05-20	2026-05-20	2026-08-18	callback.sstic.org	callback.sstic.org	C=US, O=Let's Encrypt, CN=R12

Certificate Transparency

CT logs can be accessed and queried publicly

- Makes certificates issuance auditable (monitor a domain for evil certificates)
- Third party services allow searching the database (crt.sh, censys, etc)
- Public APIs can be used to pull raw data directly from logs

Certificate Transparency

```
$ curl https://ct.googleapis.com/logs/us1/argon2026h1/ct/v1/get-sth
{
  "tree_size": 2773663726,
  "timestamp": 1777447183428,
  "sha256_root_hash": "nL9+hV5hIegbpl+OUb0tEZKM8cMc9Lp28UMHqpFNYwE=",
  "tree_head_signature": "BAMARjBEAiA…"
}
```

Get the root of the log's MT

Size of the tree (number of certificates)

Tree signature verification data

```
$ curl
'https://ct.googleapis.com/logs/us1/argon2026h1/ct/v1/get-entries?start=0&end=0'
{
  "entries": [
    {
      "leaf_input": "AAAAAAGQETJ...",
      "extra_data": "AAujAAXMMIIFyDCCA..."
    }
```

Retrieve log entries

Actual certificate logged in the tree leaf

Certificate Transparency

CT is a game changer for private key ownership attribution

- A list of public key & identities
- Ready to be mapped to leaked private keys

Mapping keys as easy as querying crt.sh a million times ?

Certificate Transparency

CT is a game changer for private key ownership attribution

- A list of public key & identities
- Ready to be mapped to leaked private keys

Mapping keys as easy as querying crt.sh a million times ? **... 502 NOPE**

Certificate Transparency

CT is a game changer for private key ownership attribution

A list of public key & identities

Ready to be mapped to leaked private keys

Mapping keys as easy as querying crt.sh a million times ? **... 502 NOPE**

Libraries and clients exists to query the logs

github.com/google/certificate-transparency-go/{client,jsonclient,scanner}

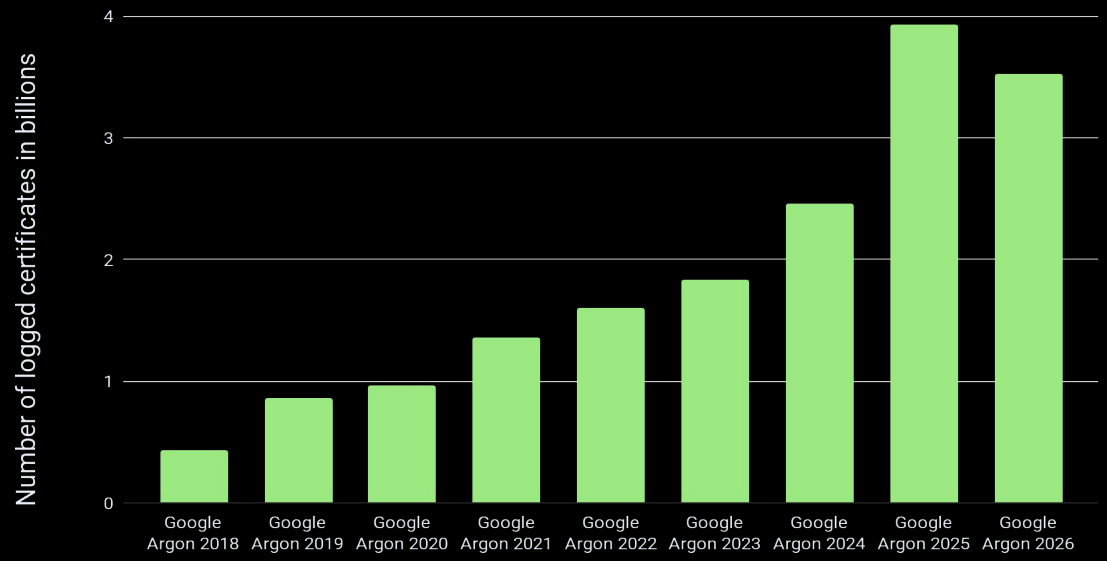

```
logClient, err := client.New(*logURL, httpClient, jsonclient.Options{})
options := scanner.DefaultScannerOptions()
scanner := scanner.NewScanner(logClient, *options)
num, err := scanner.ScanLog(ctx, CertCallback, PrecertCallback)
```

Certificate Transparency Limitations

Currently too big and too slow for us

38TB of certs for 2025 only

Rate limiting on operators' side means the download is slow



Certificate Transparency Limitations

Currently too big and too slow for us

38TB of certs for 2025 only

Rate limiting on operators' side means the download is slow

In July 2025, old logs were expected to be put offline

Don't keep CT logs online if the included certificates expired

Partnered with Google

Shortcut to retrieve all certificates of interest at once

02

Research Results

Charts, numbers & insights

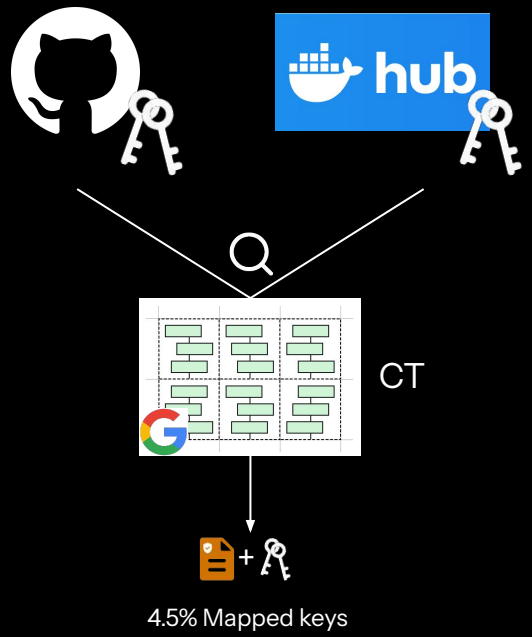
Origin of Private Keys Leaks

945,560 unique private keys
Extracted in July 2025, dated back to 2021

42,690 keys attributed thanks to CT Logs
77% from GitHub - 23% from DockerHub

Strong dataset bias
Keys are not only used for TLS, but SSH, JWT...

139,767 unique certificates
Corresponding to 36,978 subjects



TLS Certificate Validity Checks

Signature

known issuers?
altered content?

Lifetime

notBefore?
notAfter?

Revocation

OCSP?
CRL?

Name Match

subject?
SAN?

A certificate is valid if all the checks pass.

Certificates Validity

2,622 valid certificates in September 2025

35% used by publicly reachable services

Revocation is barely used

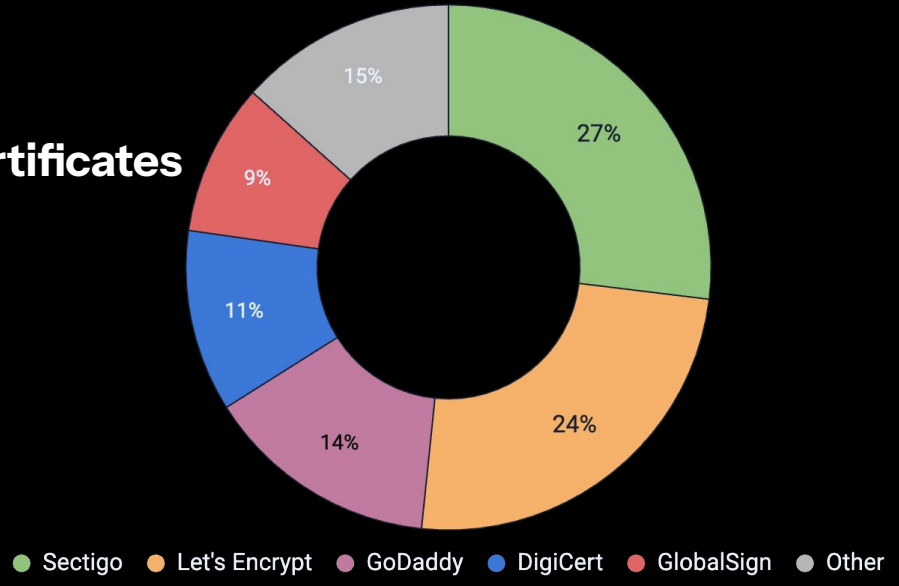
CRL: 24 invalid certificates; 1 with *key_compromise*

OCSP: 56 invalid certificates; 2 with *keyCompromise*

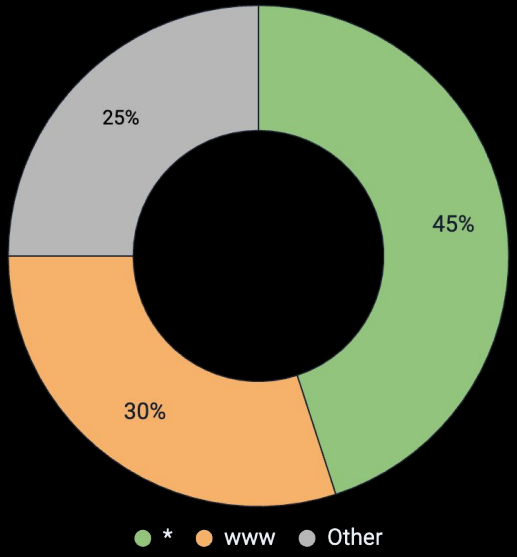
Certificate Authority Analysis

Top 5 authorities account for 85% of the certificates

Let's Encrypt is second



Hostname / Stems Analysis



Not only are free and cheap certificates being leaked, but so are wildcard certificates, which are **usually more expensive**.

Durations of Exposures on GitHub

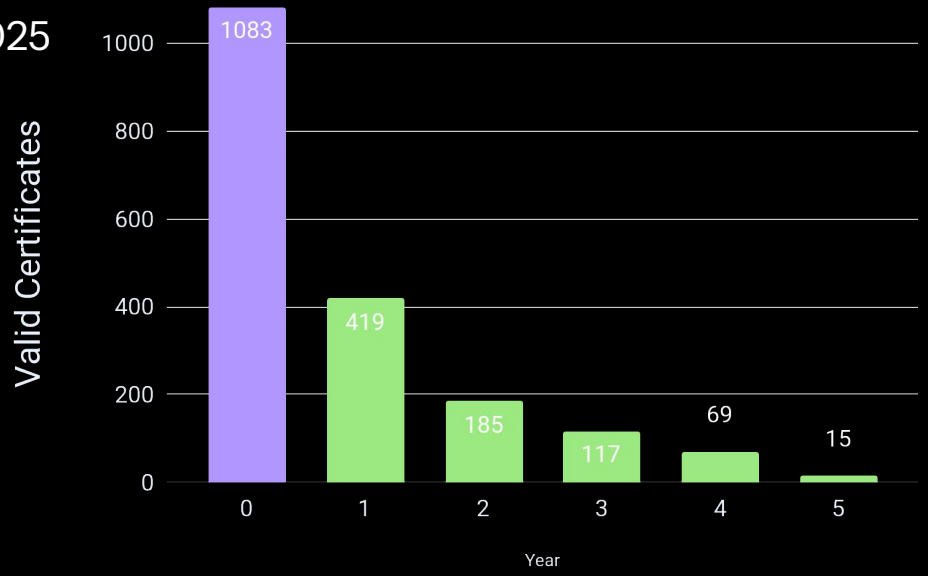
GitHub metadata allows to measure exposures

Time between first public exposure and certificate expiration

Recent vs. long-term exposure

43% of valid certificates' keys leaked before 2025

While 20% have been exposed for 2+ years



Validity at Initial Exposure

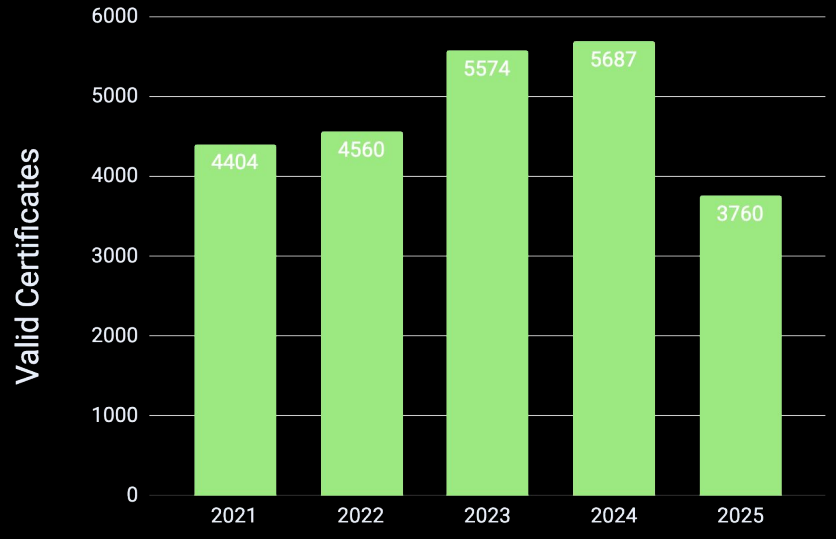
Simulating validity at leak time

Acknowledging that revocation rarely used

Compare the leak date against the certificate's validity period

Thousands of valid certificates exposed every year

17% of the 139,767 certificates discovered in CT



Fascinating Hostnames in Expired Certificates

Worldwide

snap.cso.att.com
asapdoss.ciostage.accenture.com
*.a10networks.com
*.amlarc.microsoft.com

France - Private

*.42.fr
*.rireetchansons.fr
rancher-dcadcx.michelin.fr
*.mev.atos.net

France - Public

www.paris-iledefrance.fr
mobiville.pole-emploi.fr
zen.pole-emploi.fr
particulier-dev.api.gouv.fr

⚠️Leaks from several years ago; they may never have been valid.

03
Disclosure Stories

Hard facts & lessons learnt

Identifying & Contacting Owners

Bias to fix root causes instead of revoking
We want to fix vulnerabilities
Owners need to known the bigger picture

Don't break production
At GitGuardian, we know that revoking breaks production

Various attributions techniques
X.509 Organizations, security.txt, Whois, MX, LLM...

600 organizations identified
1,300 certificates

4,300 emails sent
9 Bug Bounty programs

various types
19 governmental entities
several Fortune 500
1 Certificate Authority

Interesting Answers

Overall poor response rate
54 answers only / 9%

Low response rate for high confidence attribution
36% for companies
10% for governmental entities

Shocking misunderstanding of the critical role private keys
PoC of impact requested, confusion between keys & certificates, or validity and usage

Interesting Answers

Final Assessment

The private RSA key associated with digest [REDACTED] and fingerprint [REDACTED] is no longer active on any live service. **The endpoint now serves a different certificate** and keypair (ZeroSSL ECC, valid Jan 2026) and only performs a 301 redirect to the corporate homepage.

Based on the presented evidence, this incident represents a historical exposure rather than an active security threat. **The previously exposed key is not in use, and there is no exploitable service behind the hostname.**

How to disclose to CAs?

Certification Policy and Certification Practice Statement (CP/CPS)

Information regarding certificate lifetime
Revocation and Suspension sections

Share a proof of ownership

Better than sharing the private key
Two mechanisms: direct contact or dedicated services

No direct communication with the owners, yet some of them **asked the origin of the leaks**.

Contacting Certificate Authorities

Contacted 9 CAs
2,193 certificates

Almost immediate revocation
Aligned with Certification Policy and Certification Practice Statement

Certificates	CA Name
1,264	Sectigo
366	GoDaddy
256	GlobalSign
153	Digicert
54	GoGetSSL
52	InCommon
22	SSL Corporation
22	Starfield Tech.
4	Harica
2,193	**Total**

04
Final Thoughts

Opportunities & TLS ecosystem evolutions

Technical resilience against users' bad behavior

Certificate lifetime reducing down to 47 days (2029)
Limit potential exposure time and mitigate revocation failures

Useless if private keys are reused
cert-bot and similar already renew keys!

For issuers: forbid key re-use at CA level?
And follow CA/B rules

20% of private keys leaked more than 2 years ago

Users' misunderstanding needs technical resilience

Complement revocation with one-time private keys
Limit leak impact for users

For users: systematic private key renewal
Generate a new private key as cert-bot does

For issuers: forbid key re-use at CA level
Enforcing the key rotation at the CA level

20% of private keys leaked more than 2 years ago

Secret detection and revocation

Secret detection as a safety net

If private keys are to be leaked, better have an alerting mechanism
Beware the severity level!

In case of leak, remediate properly

Revocation is absolutely mandatory!
Replacing the certificate with a new one is insufficient
Erasing the leak source too

Certificate Transparency Evolution

Static CT

Complete rewrite of the CT API that relies on static files to serve the MT entries
Offloads the log processing to a file server (S3 compatible storage)
Cheaper for operators, much faster for consumers

```
$ curl https://mon.sycamore.ct.letsencrypt.org/2027h1/tile/data/001 | xxd
00000000: 0000 0199 0616 3c5d 0001 e376 8900 3073  ......<]...v..0s
00000010: a0c6 49cc 656d e946 c031 74d2 5c56 6fe3  ..I.em.F.1t.\Vo.
[...]
00000350: 0008 0000 0500 0000 0100 0005 5030 8205  ............P0..
00000360: 4c30 8203 34a0 0302 0102 0209 00d1 2d2a  L0..4.........-*
00000370: 69be 33a0 4830 0d06 092a 8648 86f7 0d01  i.3.H0...*.H....
00000380: 010b 0500 307f 310b 3009 0603 5504 0613  ....0.1.0...U...
00000390: 0247 4231 0f30 0d06 0355 0408 0c06 4c6f  .GB1.0...U....Lo
000003a0: 6e64 6f6e 3117 3015 0603 5504 0a0c 0e47  ndon1.0...U....G
000003b0: 6f6f 676c 6520 554b 204c 7464 2e31 2130  oogle UK Ltd.1!0
000003c0: 1f06 0355 040b 0c18 4365 7274 6966 6963  ...U....Certific
000003d0: 6174 6520 5472 616e 7370 6172 656e 6379  ate Transparency
```

Monitoring endpoint now distinct from submission endpoint

Data returned is a “tile” containing a part of the log MT

Certificates can be parsed from the tile data

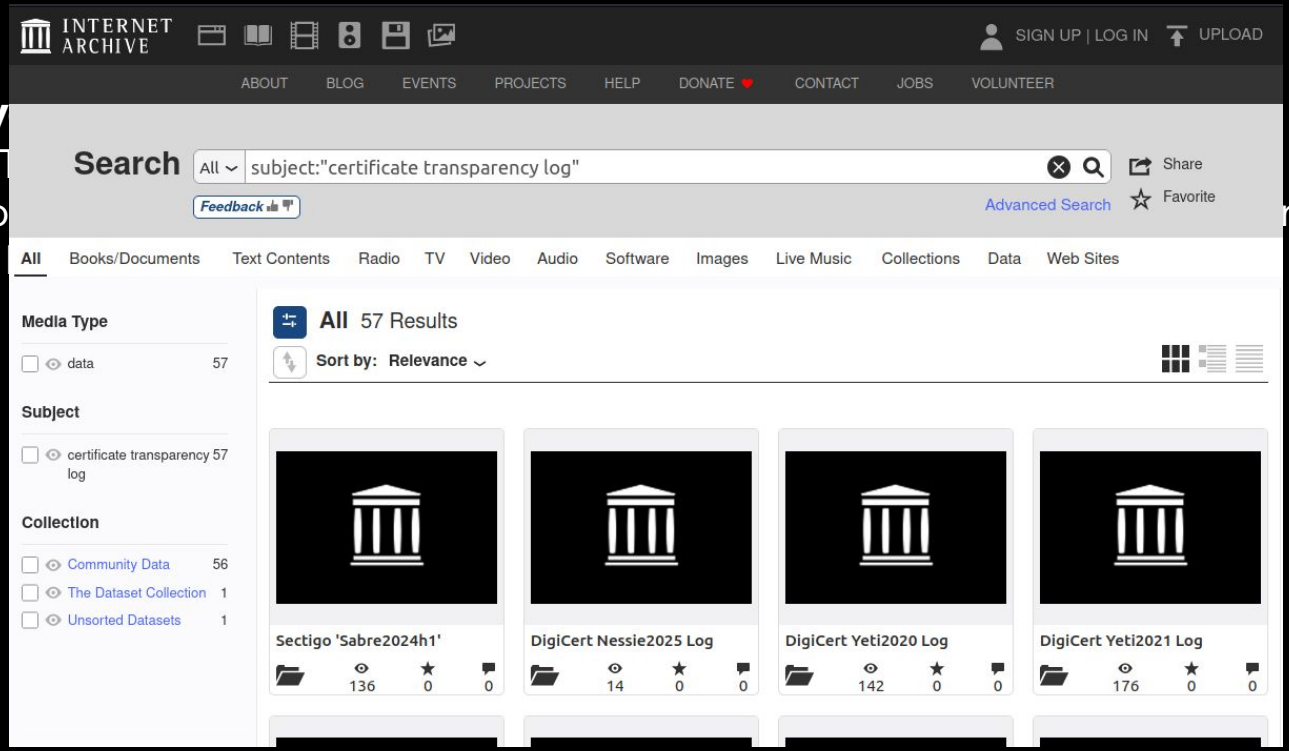
Certificate Transparency Evolution

Logs Archiv

Clone CT

Open So rchive

Years of



Take-Away Messages

Hard-facts

1. 2,622 valid certificates in september 2025
2. 20.44% have been exposed for 2+ years
3. low response rates to disclosures

Several opportunities for improvements

1. Let’s force renew private keys
2. Scan your commits for secrets

Google's colleagues enabled this research, and we encourage researchers with similar use cases to get in touch with us.

TLS Ecosystem Evolution

OCSP deprecation
Operational cost, privacy issues, cache duration…

CRLite
Compact database of revoked certificates

18.5% revoked

3% still valid
84 certificates

CRLite result on January 2026

Thank you

Question time 🔥

Philippe BONEFF - Gaëtan FERRY - David TAO - Guillaume VALADON

Improving leaked keys visibility

Private keys leak, we see it, the information does not bubble up

- Leaked keys are reused
- Some leaked keys start to be used long after
- No easy way to track leaked keys

Compromised private keys logs as a way to allow keys observability

- Similar to CT: store leaked keys information in an auditable log

Proactive prevention at issuance

- Enable CAs to check for leaked keys, and blocking re-use
- Also works for cross CA private key blacklist