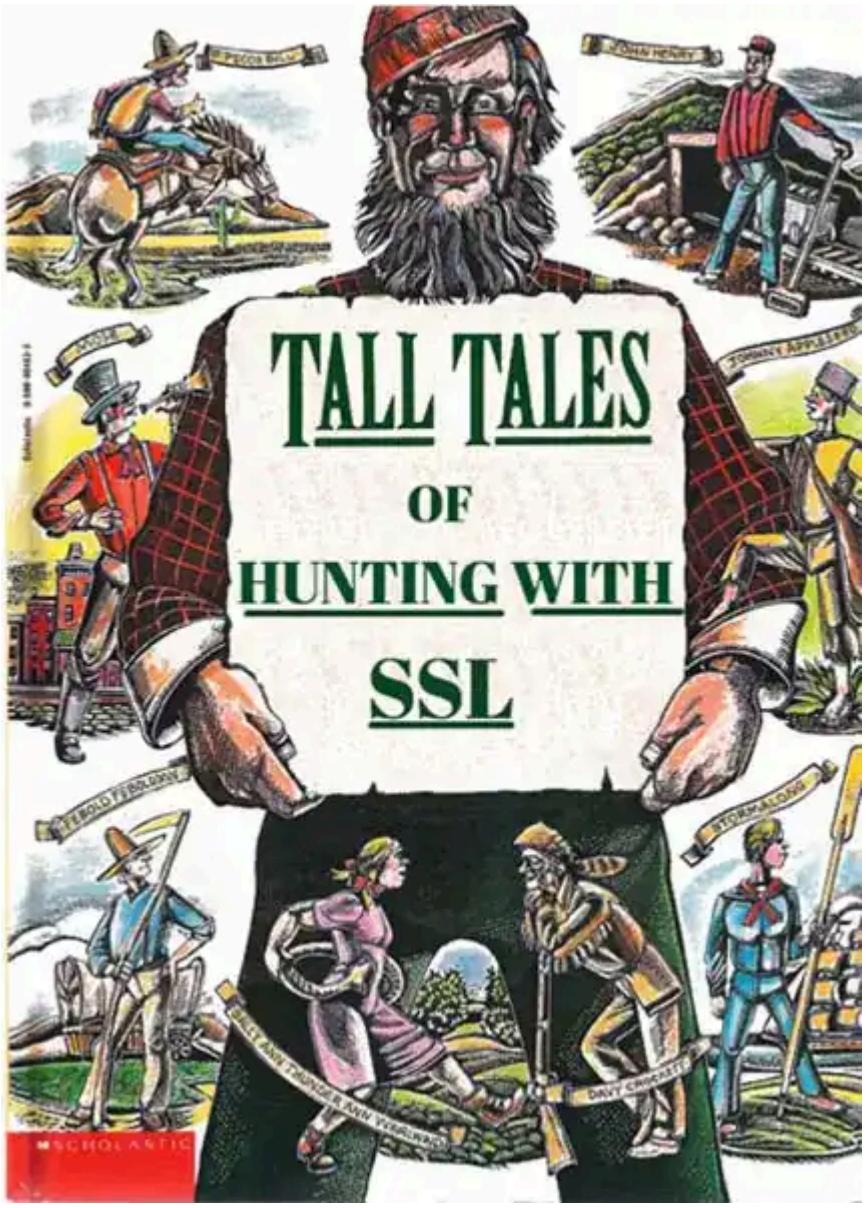
Threat Hunting with TLS/SSL Certificates | Splunk

By Lauren Stemler

Published: 2025-07-03 · Archived: 2026-04-05 22:26:06 UTC



In this article, we'll analyze how threat actors exploit TLS to hide their operations and how defenders can use exposed certificate metadata to detect them. We will discuss:

1. Why TLS certificate hunting matters
2. Which metadata fields remain visible
3. How Splunk can be used to identify suspicious TLS activity
4. Queries to enhance threat detection
5. Next steps for integrating TLS analysis into security operations.

Let’s get started!

(This article is part of our Threat Hunting with Splunk series. We’ve updated it recently to maximize your value.)

The importance of hunting in TLS certificates

First things first: this discussion will not delve into the cryptographic intricacies of TLS and SSL. Instead, it’s important to note that TLS (Transport Layer Security) is the modern, more secure successor to SSL (Secure Sockets Layer). For simplicity, the terms "SSL" and "TLS" will be used interchangeably throughout this article.

TLS has become today’s standard for secure online communications, offering stronger encryption and enhanced security protocols. However, attackers can exploit TLS to conceal their malicious activities. While TLS effectively encrypts the content of communications, it does not entirely obscure metadata, such as the identities of the communicating parties.

As cyber threats evolve, attackers have adapted to advances in security. A decade ago, malware command-and-control (C2) traffic often relied on plaintext HTTP, making it straightforward for security teams to intercept and analyze. Today, attackers leverage TLS encryption to mask their traffic, complicating efforts to monitor and detect their activities.

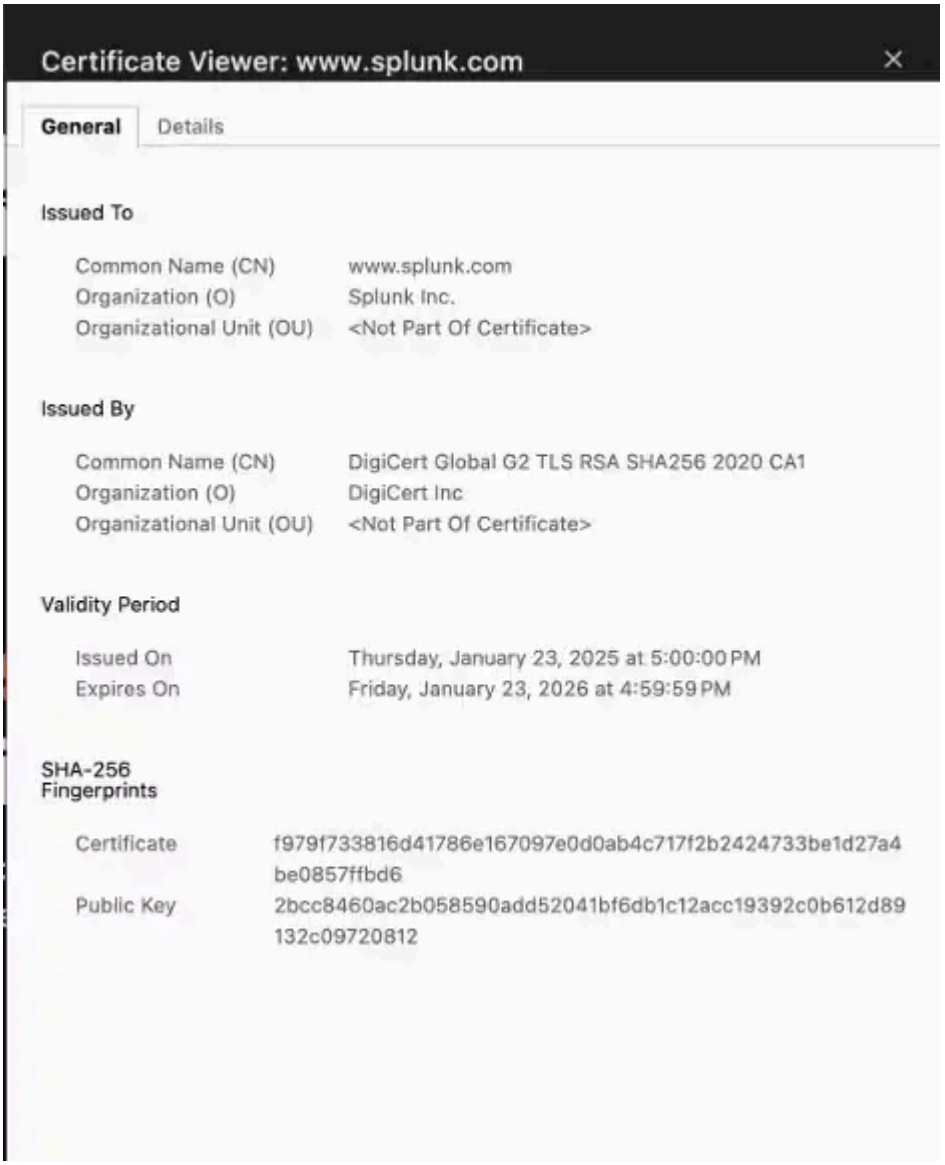
Despite these challenges, defenders are not without options. After all, we know that attackers are creatures of habit, they reuse infrastructure and leave data behind that defenders can use to trace their actions back to them. By identifying these clues, security teams can uncover malicious activity and link it back to its source.

(Note: This article assumes that you have already captured TLS metadata in Splunk. If not, implementing metadata collection will need to be done before diving into these hunting techniques.)

What unencrypted information does SSL leave?

TLS certificates are like digital ID cards for websites and services, proving their authenticity while enabling encrypted information. But here’s the catch — while the actual communication is encrypted, the certificate itself contains unencrypted metadata that attackers can’t hide. Defenders like you can use this metadata to identify and track suspicious activity.

The fields within a TLS certificate are generated either by cryptographic algorithms or manually entered during the creation of the certificate.



How does this help defenders? Because these fields can be unique and are often shared by adversaries across their backend infrastructure. So… what are some unique fields? The ones that we are primarily interested in are:

- `ssl_cert_self_signed`
- `ssl_subject_country`
- `ssl_subject_state`
- `ssl_issuer`
- `ssl_subject_locality`
- `ssl_subject_unit`
- `ssl_subject_organization`
- `ssl_validity_end`
- `ssl_subject_common_name`
- `ssl_serialnumber`
- `ssl_cert_sha1`

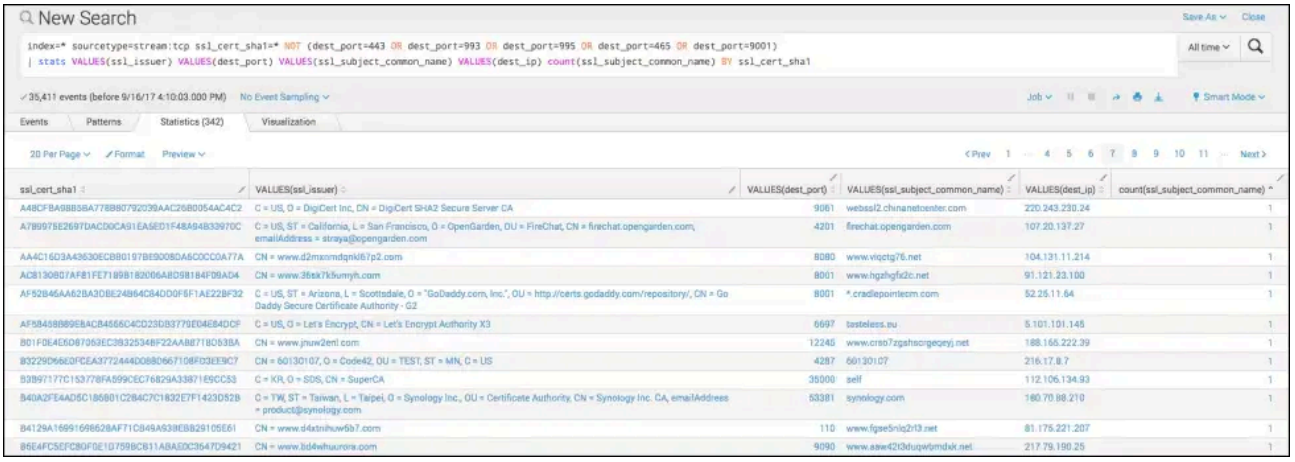
These fields can help security teams detect anomalies, track threat actors, and distinguish legitimate infrastructure from adversarial activity.

Now that we know TLS certificates leave behind valuable metadata, how do we actually use this information? Since adversaries are using TLS to encrypt command and control (C2) communications, it's important to look beyond the encryption itself and analyze the exposed metadata. Since we have a hypothesis that malware is beginning to use SSL to encrypt its communications, I believe it is also reasonable to assume that they would use SSL to encrypt communication on alternative ports!

Finding encrypted communications in SSL

For nearly all the following examples, we will be using SSL data captured in Stream. Let's look:

```
sourcetype=stream:tcp ssl_cert_sha1=* NOT (dest_port=443 OR dest_port=993 OR dest_port=995 OR dest_port=465 OR
| stats VALUES(ssl_issuer) VALUES(dest_port) VALUES(ssl_subject_common_name) VALUES(dest_ip) count(ssl_subject_
```

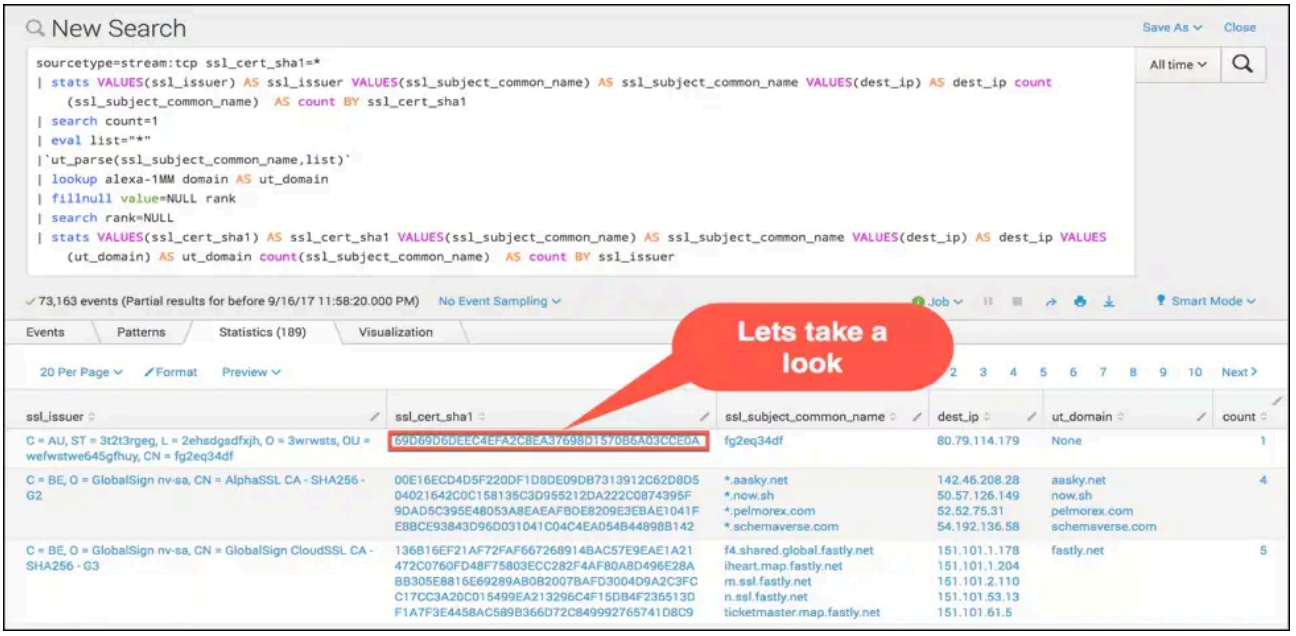


BOOM! You'll notice the 3rd and 4th entry have very suspicious metadata and link to some nasty looking domains and IP addresses. A quick pivot to VirusTotal shows that there have been malicious files hosted on that IP address in the past:

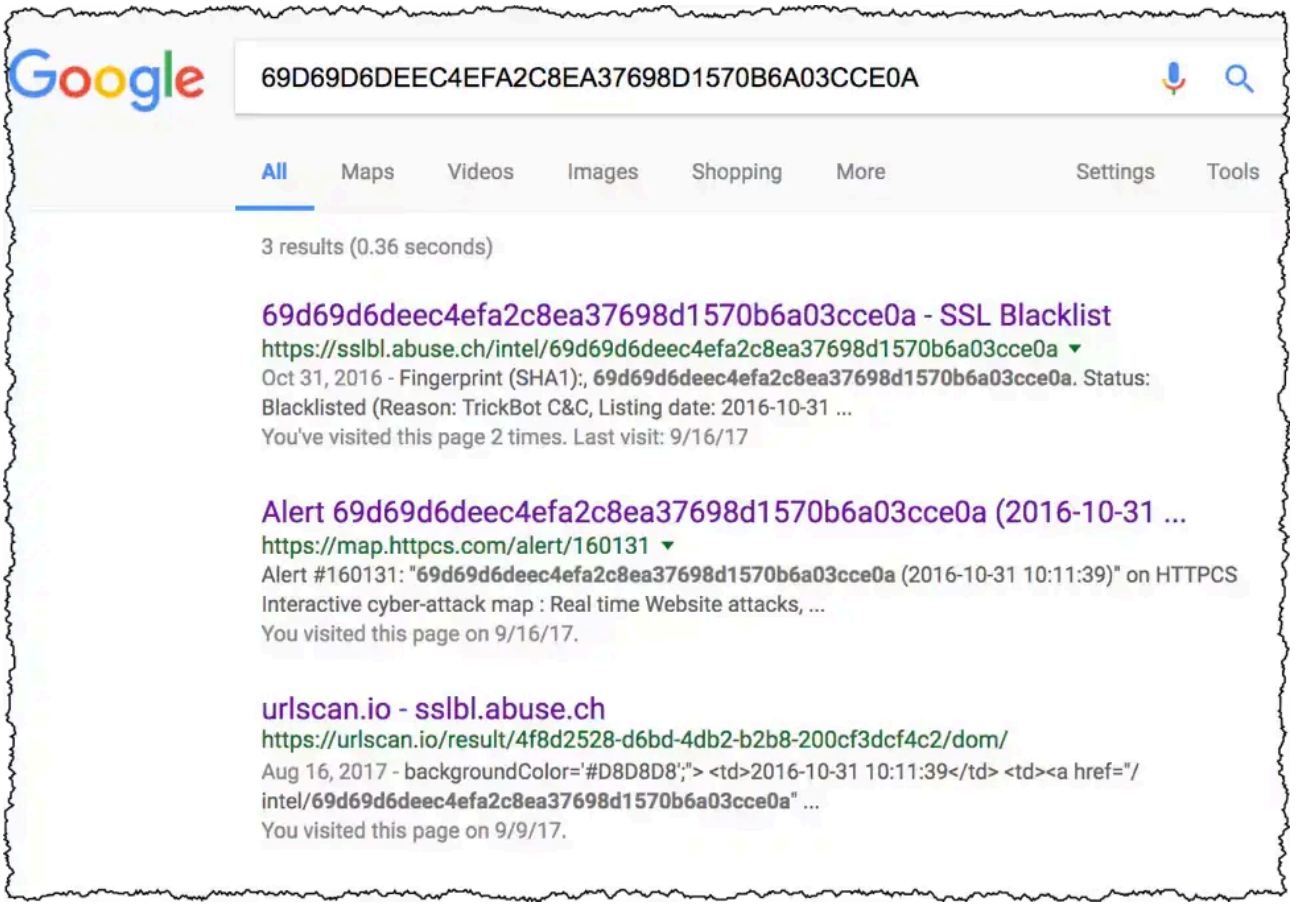
Finding unique values in SSL certificates

For our next search, let's try and find values in SSL certificates that are unique. We will do a little pre-parsing by showing only SSL certificates that are not part of the Alexa 1 Million top sites. This is done with the UT_PARSE macro (which you can install with URL Toolbox) and the Alexa 1 Million list. You can download a sample of Alexa 1 Million from this GitHub as a properly formatted lookup table.

```
sourcetype=stream:tcp ssl_cert_sha1=*
| stats VALUES(ssl_issuer) AS ssl_issuer VALUES(ssl_subject_common_name) AS ssl_subject_common_name VALUES(dest
| search count=1
| eval list="*"
|`ut_parse(ssl_subject_common_name,list)`
| lookup alexa-1MM domain AS ut_domain
| fillnull value=NULL rank
| search rank=NULL
| stats VALUES(ssl_cert_sha1) VALUES(ssl_subject_common_name) AS ssl_subject_common_name VALUES(dest_ip) AS des
```



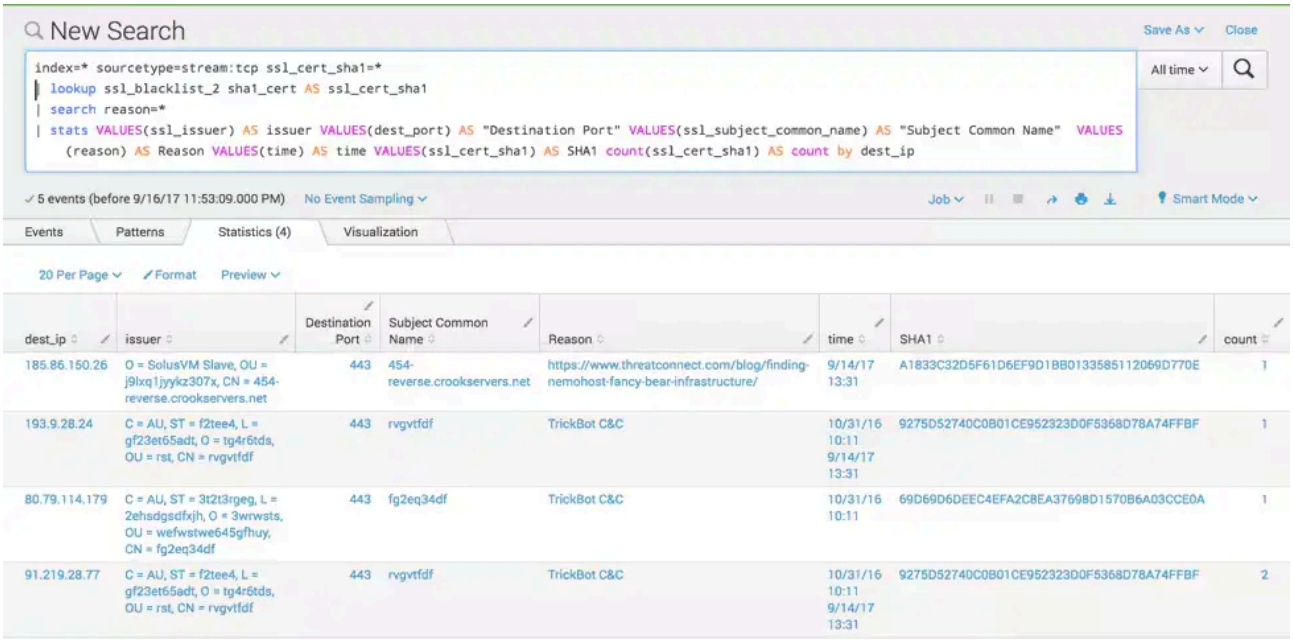
Right away we see some very odd values in the `ssl_issuer` and `ssl_subject_common_name` fields. If we search for that SHA1 certificate value (ssl_cert_sha1), we find that various websites have identified that SHA1 hash as malicious.



Detecting bad SSL certificates

Finally, what about detecting certificates that are “known bad”? The most common repository of externally identified certificates is from https://sslbl.abuse.ch/. With minimal effort, you can turn that list of malicious SHA1 SSL certificates into a lookup table in Splunk. The example below assumes that you have done that already, but if you want an example, visit my GitHub and download the ssl_blacklist_2.csv file.

```
index=* sourcetype=stream:tcp ssl_cert_sha1=*
| lookup ssl_block_list sha1_cert AS ssl_cert_sha1
| search reason=*
| stats VALUES(ssl_issuer) AS issuer VALUES(dest_port) AS "Destination Port" VALUES(ssl_subject_common_name) AS
```



The results below show that we have five hits against our data and a quick explanation of each malicious certificate.

Next steps in security operations

Finding malicious SSL/TLS in your environment is just the first step. Just like finding malicious IP addresses or file hashes you will need to pivot and research using tools like

- www.passivetotal.com
- www.censys.io
- https://www.circl.lu/services/passive-ssl/

Once you understand how to use these certificates, try creating workflow actions in Splunk to pivot “automagically”.

If you’d like an example, check out my sample workflow action for www.censys.io located on my github page. If you are interested in more information on hunting and pivoting with TLS certificates, check out this great talk from my good friend Mark Parsons on hunting a TLS certificate. Until then…

Happy hunting!
