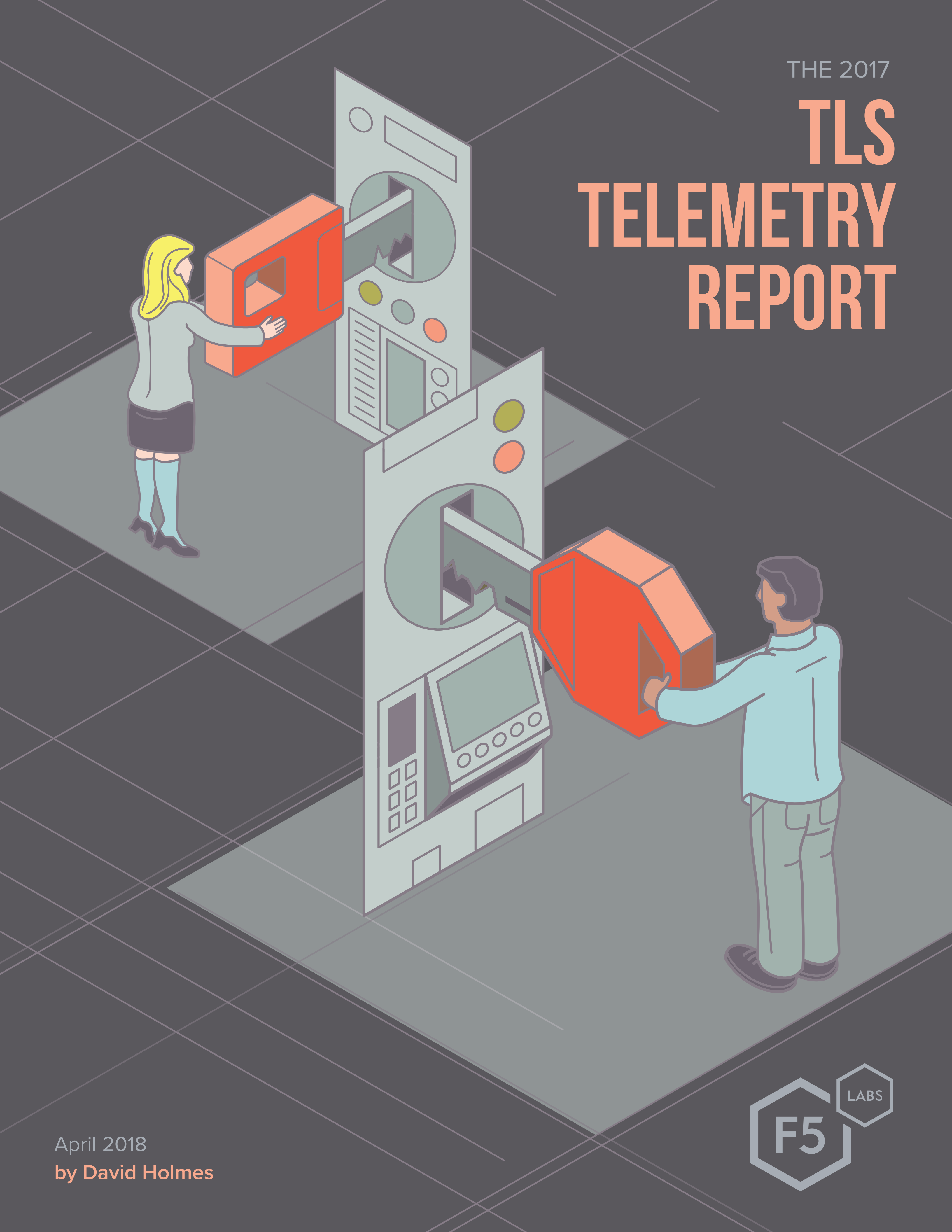
THE 2017

TLS TELEMETRY REPORT



April 2018

by David Holmes

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

EXECUTIVE SUMMARY

Privacy today isn't just about staying away from prying eyes. The very act of communicating across the Internet with open, non-confidential protocols invites exposure to code injections, ad injections, and overall risk injection.

A recent example involved spyware that was developed by FinFisher and only sold to nation-states. Middleware systems inside the Türk Telekom IPS hijacked unencrypted web sessions, then redirected the sessions to websites that installed the spyware. It appears this was a targeted attack on five provinces in Turkey and specific locations in Syria. Similar middleware systems were also used inside of Telecom Egypt to, again, hijack unencrypted web sessions. This time the motive was financial: the redirects sent unsuspecting users to revenue-generating ad farms and crypto-mining malware downloads.[i]

This is why cryptographic protocols like Transport Layer Security (TLS) exist—to help prevent adversaries from eavesdropping and tampering with data. TLS provides a way for endpoints to be authenticated and communicate confidentially over the Internet. Encrypting Internet connections with TLS would have stopped these redirect attacks. But like all security protocols, it must be constantly tested, proved, and improved if we have any hope of staying ahead of adversaries.

F5 has been diligently monitoring the "cryptographic health" of the Internet since 2014. In 2016, we began reporting on the state of TLS in the F5 Labs 2016 TLS Telemetry Report. With the benefit of nearly four years of data, we've seen some positive signs of progress and some lingering areas of concern. In this, our second report in the series, we share our key findings for 2017, based on a sampling of more than 20 million SSL/TLS hosts worldwide:

- TLS's predecessor, SSL 3.0—which is now prohibited from use by the Internet Engineering Task Force (IETF)—is taking its time disappearing entirely from the Internet. 11.2% of Internet hosts still support it.
- The transition from TLS 1.1 to TLS 1.2 has been steady, with 27% more hosts making the move in 2017. Currently, 89% of hosts are using TLS 1.2.
- IETF's progress on TLS 1.3 has been slow for many reasons, not the least of which is debate about whether TLS 1.2 is really "broken" enough to require fixing.
- The HTTP Strict Transport Security (HSTS) header, important because it instructs the browser to always use a secure connection, is finally seeing some forward motion, even though numbers are still very low. Since the summer of 2014, HSTS usage has grown from a mere .33% to just over 4% in Q1 of 2018.
- Forward secrecy, a cryptographic technique designed to prevent adversaries in the future from decrypting captured, encrypted sessions from today, is steadily being adopted. Now, 88% of hosts prefer forward secrecy, up from about 30% in 2014.
- Self-signed certificates (those not signed by a trusted Certificate Authority) dropped from 15.2% in the first quarter of 2017 to 11.6% in Q1 of 2018. At the same time, free certifications from Let's Encrypt jumped so sharply in 2017 that it is now the second most popular Certificate Authority website.

Security teams can use these findings to evaluate their security posture and make improvements where needed. We've also included recommendations around the use of HTTPS for every website as well as the use of HTTP Strict Transport Security (HSTS); the use of Online Certificate Status Protocol (OCSP) stapling to ensure the validity of digital certificates; the need to patch to protect devices from ROBOT attacks; and the recommended use of Qualys SSL Labs to test your website's cryptographic security posture.

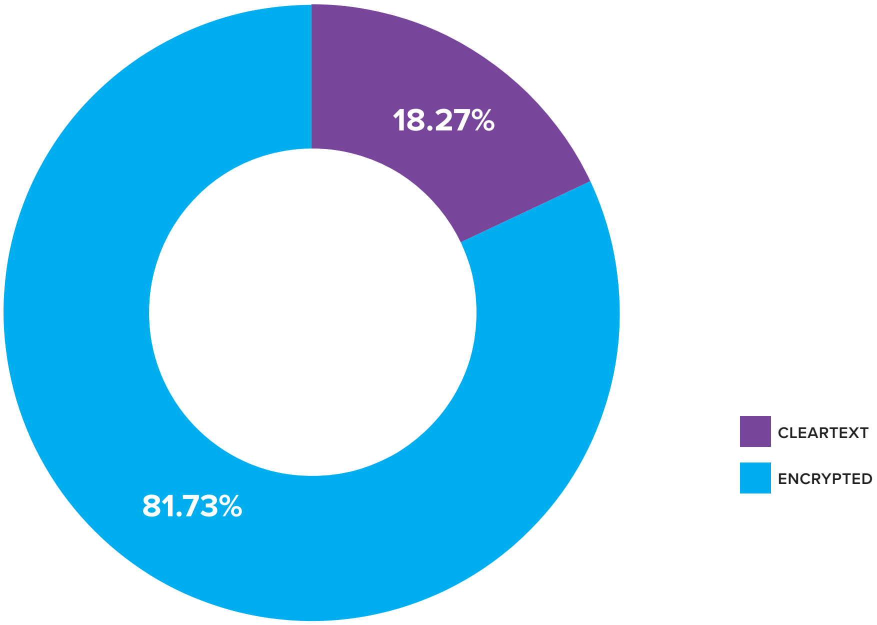
INTRODUCTION

In the F5 Labs 2016 TLS Telemetry Report, we introduced you to the core cryptographic statistics across the entire Internet, which we had sampled for three years (2014 through 2016). In the 2017 edition of this report, we'll show you what's changed, what's new, and make some predictions about where everything is going in the world of encryption. Our current focus on the statistics of preferred cryptographic protocols and ciphers among HTTPS servers is more relevant than ever. Over *80% of page loads* are now encrypted with SSL/TLS, according to Firefox's metrics dashboard. Look at that statistic again: TLS is now *the most important protocol on the Internet*.

FIGURE 1

ENCRYPTED HTTP PAGE LOADS

source: telemetry.mozilla.org



We've continued building out our scanning capabilities. It hasn't been easy, because the trend in the open source world is to remove cryptographically weak ciphers, and that makes it difficult to develop a toolchain that can still scan for old ciphers like SSL 3.

OVER 80% OF PAGE LOADS ARE NOW ENCRYPTED WITH SSL/TLS.

Note: *Before we dive into the details, it's important to point out that, unlike other F5 Labs reports that take a completely vendor-neutral stance, this report includes data about F5 usage across the Internet for the simple fact that F5 devices decrypt a statistically significant amount of Internet traffic. This information is in no way intended to be promotional. F5 Labs is committed to providing completely vendor-neutral, application-related threat intelligence to the general public. We've included the details of our scanning methodology in the appendix of this document.*

CORE CRYPTOGRAPHIC STATISTICS

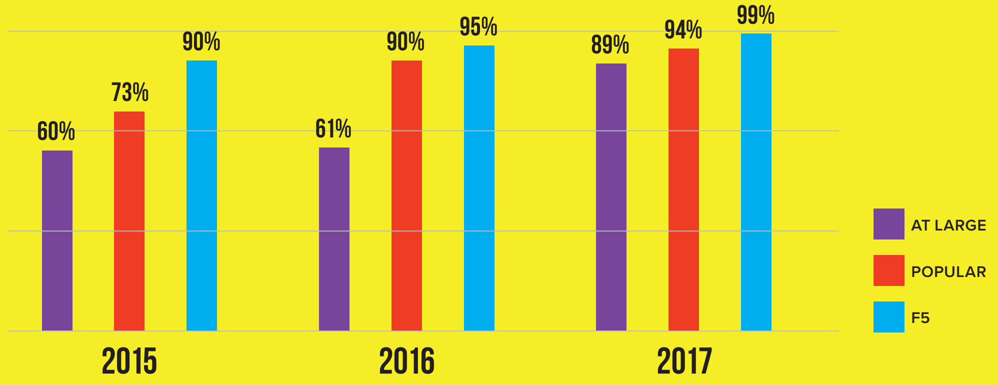
The first step in establishing an SSL/TLS connection is the negotiation of the protocol version. Sometimes it can be a pretty awkward dance, at least for the poor browsers that have to try to connect as quickly and securely as possible while still supporting ancient, non-compliant hosts. The preferred protocol version is the broadest indicator of cryptographic posture of hosts across the Internet.

TLS 1.2 IS PEAKING

During the five quarters that have passed since we wrote the 2016 TLS Report, we sampled 20 million SSL/TLS hosts (20,321,901 to be exact) and recorded their preferred protocol. In the majority of cases, their preferred protocol is typically the highest, most secure version of TLS.

FIGURE 2

PREFERENCE FOR TLS 1.2 SINCE 2015



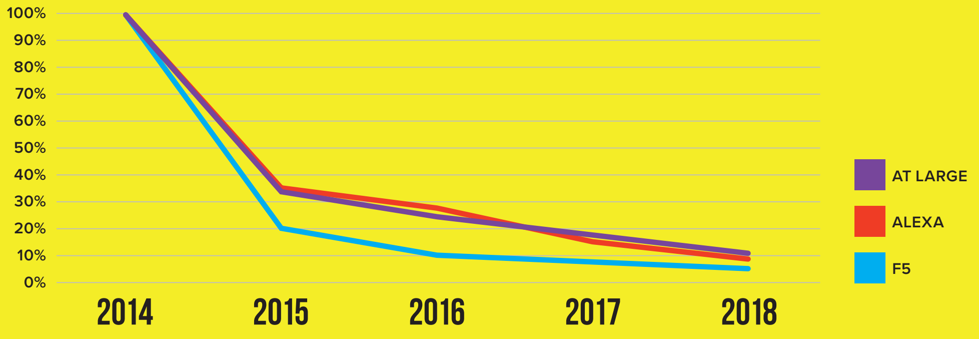
The good news is that we've seen a massive number of hosts move from TLS 1.0 to TLS 1.2. At the end of 2016, preference for TLS 1.2 was 62%. At the beginning of 2018 it had risen to 89%.

SSL 3 IS DYING A SLOW DEATH

As expected, support for SSL 3 continues to dwindle, although slowly. Plunging from 98% support to less than 40% after the POODLE vulnerability in late 2014, SSL 3 is taking longer to die than we might have expected. At its current rate, it could be years before it's finally gone.

FIGURE 3

SSL 3 SUPPORT DROPPED DRAMATICALLY FOLLOWING THE POODLE VULNERABILITY OF 2014



TLS 1.3 IS STUMBLING OUT OF THE GATE

The design of version 1.3 of the TLS protocol was contentious from the beginning. First of all, for most architects, there was nothing really wrong with TLS 1.2 from a cryptographic perspective. Thus, the design priorities for TLS 1.3 were somewhat muddy. Early drafts of the protocol tried to solve information leakage problems. For example, the TLS client has to send the “ServerNameIndicator” (SNI) in cleartext. In theory, this would enable a snooper (an oppressive nation-state, perhaps) to determine which freedom blogs a possibly seditious citizen was attempting to connect to. The Internet Engineering Task Force (IETF) TLS committee spent many hours trying to solve this problem but, honestly, it’s not solvable, at least without some pretty serious changes to DNS (been there, tried that).[ii]

For Google, TLS 1.2 wasn’t happening quickly enough, so the company started building TLS-like protocol mutations years ago.[iii] Eventually, after some experimentation, Google built and adopted its own protocol, QUIC.[iv] It has lower latency than TLS 1.2 and supports tricks like sending data on the first packet (TLS 1.2 does not). It comprises much of the UDP traffic that our service provider friends are seeing.

The IETF TLS committee then had to play catch-up to Google and make a TLS protocol that would work for everyone. Complicating things even further, parts of the existing TLS protocol, such as non-forward-secret ciphers and artifacts like the “ChangeCipherSpec” message, were deprecated. These protocol changes were impactful enough that many devices do not recognize TLS 1.3 as the TLS protocol and are choking on it. That, and the fact that there isn’t a compelling security (vulnerability) reason to abandon TLS 1.2, has stalled adoption of TLS 1.3.

We can detect hosts that prefer TLS 1.3, but honestly, outside of Cloudflare, almost no one is using it. Out of a sample of nearly 5 million websites in January 2018, excluding Cloudflare, a mere 3,400 (0.07%) prefer TLS 1.3.

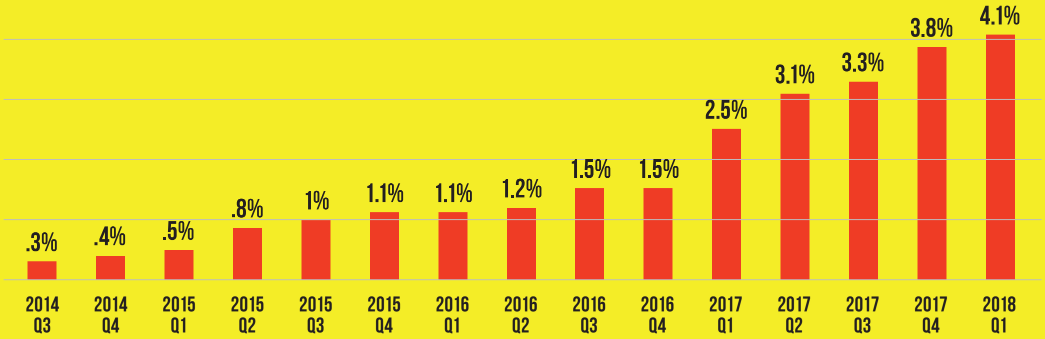
The protocol designers have encountered some of these problems and are now having to go back and rework the protocol. But without a compelling security reason to upgrade, ultimately TLS 1.3 may be no more successful than TLS 1.1, which no one ever preferred.

HTTP STRICT TRANSPORT SECURITY IS FINALLY TAKING OFF

After *years* of languishing below 5%, one of the most important HTTP security headers is finally getting some traction. HSTS, the HTTP “Strict Transport Security” header, instructs the browser to always use a secure connection when it returns to a website. Sounds like a no-brainer, right? But HSTS has three main gotchas. The first being that, to be useful, the directive should apply to all associated subdomains, as well. Many applications had subdomain services that weren’t considered important enough to have an associated certificate and HTTPS channel. That has mostly gone away as people have realized that without HTTPS protection, service providers and anyone else with access to your HTTP can inject all kinds of funky code. Second, HSTS has a six-month horizon, so if administrators had set HSTS but weren’t actually ready for it, they could have blackholed their site. Lastly, millions of static, mom-and-pop websites are served by big hosting providers, which have little interest in securing them after the fact.

FIGURE 4

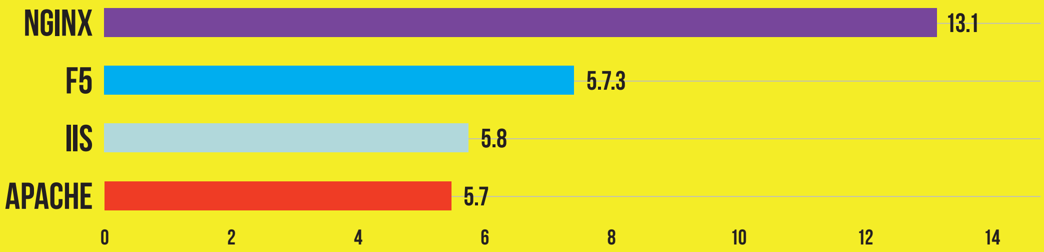
PREVALENCE OF HTTP STRICT TRANSPORT SECURITY (HSTS)



When we started our research in the summer of 2014, overall usage of HSTS was 0.33%. That’s right, just a third of one percent. Today, overall, it’s about 4%—with the majority of the growth coming within 2017.

FIGURE 5

USE OF HTTP STRICT TRANSPORT SECURITY (HSTS) ACROSS VARIOUS TLS STACKS



Among F5 users, the statistic was even worse in 2014. Less than 0.1% of F5 TLS servers use HSTS. Today, 7.3% of F5 devices require strict transport security.

We see these as good trends for Internet security.

FORWARD SECRECY IS CHARGING FORWARD

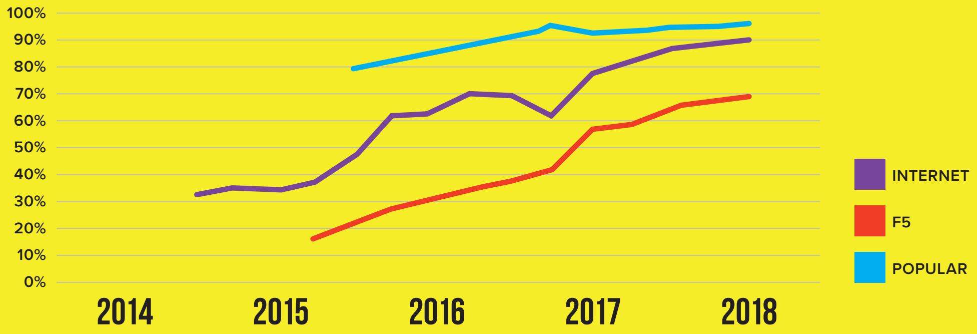
The most striking, and relevant, of all the statistics in our reports is the rise of forward secrecy. Forward secrecy is the cryptographic technique of foiling passive surveillance. Imagine that you are regularly communicating with a server in Elbonia—enough that it gets the attention of your government and they want to know exactly what information you are sending and receiving from there. They try to tap your communication, but you're using TLS with decent-sized public keys (say, 2048-bit RSA), so they can't crack it. Instead, they record your encrypted sessions. Fast-forward a year, and they get an operative to Elbonia to locate the server and steal the RSA key. Now they have the ability to decrypt all of your previously recorded traffic.

Forward secrecy solves that problem by adding an ephemeral key exchange on top of the RSA key exchange such that only the two endpoints, at that time, can decrypt each other's traffic.

When we started our original analysis in 2014, forward secrecy was preferred by a third of randomly-sampled Internet hosts. Today, it is nearly ubiquitous: 88% of hosts prefer forward secrecy.

FIGURE 6

FORWARD SECRECY AS A PERCENTAGE OF POPULATION



Very likely, the reason it's so high is because forward secrecy is the preferred default for the major SSL stacks and, of course, in the OpenSSL library.

TODAY, FORWARD SECRECY IS NEARLY UBIQUITOUS: 88% OF HOSTS PREFER FORWARD SECRECY.

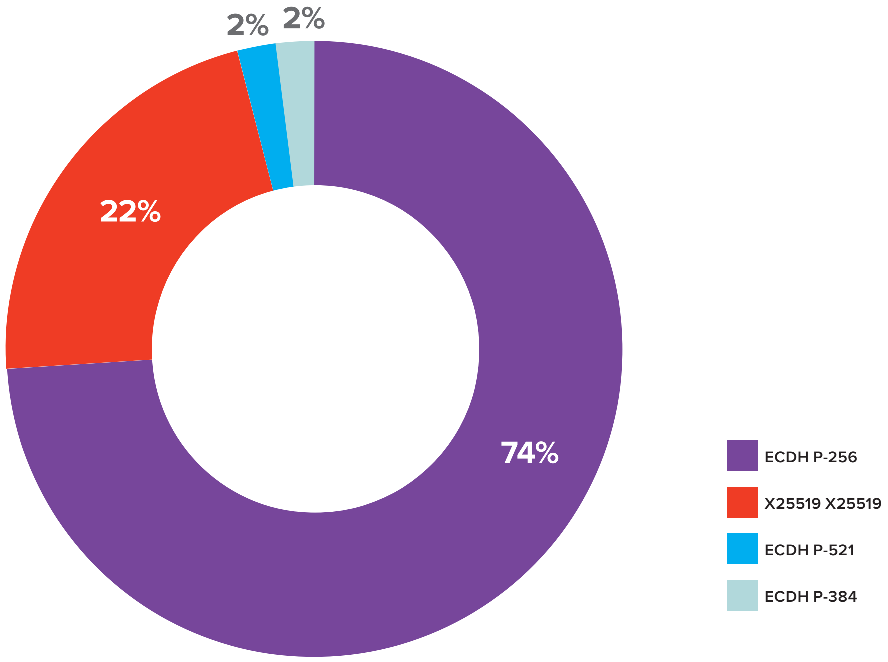
COMPARING THE MANY FLAVORS OF FORWARD SECRECY

Perhaps the most exciting research we've added in the 2017 TLS Telemetry report involves the popularity of the different flavors of forward secrecy. When we first started our research in 2014, we could only differentiate three types of key exchanges: RSA, Diffie-Hellman Emphemeral (DHE), and Elliptic Curve Diffie-Hellman Ephemeral (ECDHE). Back then, RSA and DHE method were the most popular. However, as we reported in the 2016 TLS Telemetry Report, elliptic curves finally took off, eventually became favored, and now dominate forward-secret key exchanges at 85%.

This is where things get interesting. A prominent chip maker asked us at F5 Labs if we could tell them exactly which curves seemed popular. Specifically, they wanted to know if anyone was using the cool new curves, X25519 and X448. So, we added that check to our most recent scan tools.

FIGURE 7

PERCENTAGE OF ELLIPTIC CURVE CRYPTOGRAPHY FLAVORS IN USE AS OF Q4 2017



NIST P-256: Today's Favorite, but Not Tomorrow's

As shown in Figure 7, the most popular (preferred) elliptic curve is NIST P-256, followed by X25519. Curve NIST P-256 is the current chart-topper. P-256 is an old Weierstrass curve, part of NIST's Suite B playlist, and is still the only curve that is supported in the FIPS 140-2 specification. You may also know P-256 as prime256v1. P-256 is the best curve for interoperability, and people have spent a lot of time optimizing and promoting it.

Three quarters of the Internet prefers P-256 to other curves, and that's probably because it's the default curve for OpenSSL, which in turn is the default TLS stack for the Internet. Even with its name recognition and broad support, P-256 might have seen its peak. P-256 has some baggage; because of its origin from NIST (and by association, the NSA), some claim that P-256 may have a cryptographic back door.

X25519 (Curve 2^{255}-19) is the Future

X25519 describes a Diffie-Hellman key exchange with the curve described by the prime number 2^{255}-19. It's in everyone's playlist right now and getting big fast. Spotify and iOS use it. It's already supported by most major browsers, TLS vendors, cloud providers, and content delivery networks, although many just support it rather than prefer it.

X25519 has a lot of things going for it. First, it is designed by Daniel J. Bernstein himself; the source code is public domain and the algorithm has no patents. Bernstein claims it is provably secure and that, by design, it's immune to known timing attacks.

X25519 is recommended by the IETF TLS committee, and most recently, NIST has also publicly stated that they are behind it.[vii] X25519 has 40% lower computational complexity than P-256. Virtual machines should prefer it to P-256. The lower complexity also means that it will be easier to process on low-CPU devices like the billions found in the Internet of Things (IoT).

Curve448: More Secure but Worth It?

Curve448 is an Edwards curve, which has simpler mathematics than Weierstrass curves and can also be used for elliptic curve digital signatures (certificates).[viii] It's not represented in a measurable form in our scanning data, but it is worthy of some discussion here, as it relates to P-384, which is currently preferred by about 2% of TLS hosts among the OpenDNS top 1 million.

P-384, also called secp384r1, is another NIST-approved Suite B elliptic curve with a larger key size (384 bits). P-384 has a performance penalty of about a factor of three over P-256, and a few dozen extra bytes on network in a handshake.

The intent of both P-384 and Curve448 is to provide higher security than that of P-256 and X25519. However, it's hard to achieve higher security because the key exchange is just one part of TLS. The higher key exchange security needs to be matched diligently throughout TLS with higher strength symmetric keys (like AES256 instead of AES128 and SHA512 instead of SHA256), etc. Whether that higher security is accomplished throughout or not, there will be a performance penalty for P-384/X448. At this time, it seems most of the Internet is content with the 128-bit security offered by P-256 and X25519.

Currently Suite B has P-384, not P-521 nor X448. Suite B is, itself, in the transition period out of ECC toward quantum-safe algorithms. But even with the Suite B stamp of approval, P-384 hasn't taken off and isn't likely to. No notable effort went into optimizing P-384, so it's a double-hit: higher complexity and fewer optimizations. When looking at these factors, if pushed to support one or the other, vendors will likely choose to do X448 in TLS rather than P-384. Like P-384, about 2% of hosts prefer P-521, which offers even higher security strength. But P-521 isn't likely to take off for the same reasons that P-384 won't.

THE INTENT OF BOTH P-384 AND CURVE448 IS TO PROVIDE HIGHER SECURITY THAN THAT OF P-256 AND X25519. HOWEVER, IT'S HARD TO ACHIEVE HIGHER SECURITY BECAUSE THE KEY EXCHANGE IS JUST ONE PART OF TLS.

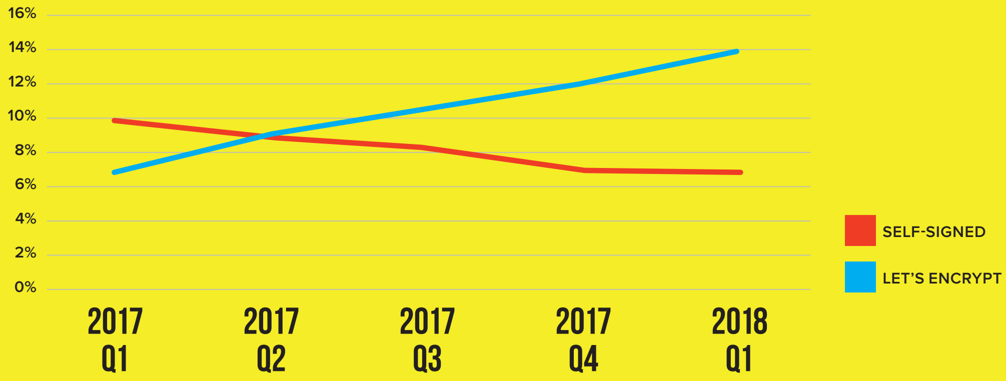
SELF-SIGNED CERTIFICATES ARE DROPPING

The public cryptographic key should be signed by a *trusted third party*. This is one of the fundamental principles of any asymmetric cryptographic system, including TLS. Typically, the trusted third party is a certificate authority such as Comodo, DigiCert, or GlobalSign. If a client lacks a trusted signature, it cannot be certain of the authenticity of the site it *believes* it is talking to. Surprisingly, a remarkable number of Internet hosts present a certificate wherein *the public key has been signed by the associated private key*. These are called self-signed certificates, and they make verification of the authenticity of the certificate nearly impossible.

Because anyone can generate a self-signed certificate at any time, for any purpose, they are inherently untrustworthy. They're also free, so there's a lot of them out there. In the 2016 TLS Telemetry Report, we decried the high percentage of self-signed certificates on the Internet at large (nearly one third). Also in 2016, Let's Encrypt, the free, open-certificate authority, was just getting off the ground and issuing certificates. In the year following publication of our initial report, we've been monitoring the rise of TLS hosts that use Let's Encrypt for certificates.

FIGURE 8

LET'S ENCRYPT VERSUS SELF-SIGNED CERTIFICATES



In general, there has been a decline in self-signed certificates and an increase in Let's Encrypt certificates. In other words, there are now twice as many Let's Encrypt certificates as there are self-signed among popular websites. Is that causation or correlation? We suspect some of each. What's probably more disturbing is that one in fifteen of the most popular sites in the world still use self-signed certificates.

AN ORDERED LIST OF CERTIFICATE AUTHORITIES

Since we're on the topic of certificate authorities, it's pretty easy to see which ones are the most popular. Whenever you connect to a host, it presents its certificate, which contains the name of the certificate authority that signed it.

In our sampling at the beginning of 2018, we totaled up the numbers for the top 10 certificate authorities. It's mind blowing that just a few years ago Let's Encrypt didn't even exist, and now it's the number two certificate authority represented among popular websites. That speaks to the power of "free."

TABLE 1

TOP 10 CERTIFICATE AUTHORITIES

COUNT	CERTIFICATE ISSUER
160,377	COMODO CA Limited
92,425	Let's Encrypt
62,998	DigiCert Inc
57,602	GeoTrust Inc.
50,961	GoDaddy.com, Inc.
44,783	Self-Signed
32,404	GlobalSign
30,639	cPanel, Inc.
21,758	Google Inc
17,031	Amazon

Comodo retains the top spot, as it has for a decade.

FUN STATISTICS

Opposite the “core” cryptographic statistics that we covered in the previous section are some that we track not so much for a sense of security posture, but just because we think they’re interesting. Or weird corner cases. Or both.

“TOP 1 MILLION” FACEOFF: ALEXA VERSUS OPENDNS

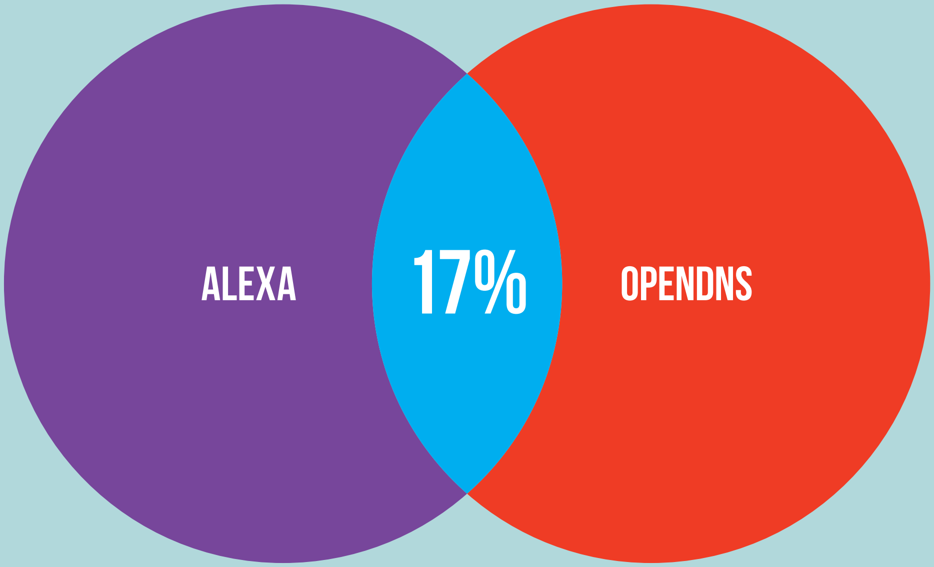
In our first TLS Telemetry report, we used the Alexa “Top 1 Million” as a primary source of hostnames. The Alexa list is maintained by a division of Amazon and relies on statistics from the Alexa browser plugin. The list is broadly cited, but many people point to inconsistencies. For example, someone might own two or more sites on the Alexa list, and therefore have access to the raw analytics, which often show that Alexa gets the ordering wrong. Also, the Alexa plugin is not widely used, even by people who run multiple search engine toolbars at the same time. But, for our purposes, the ordering of the list is not important; we just care about stats and hostnames.

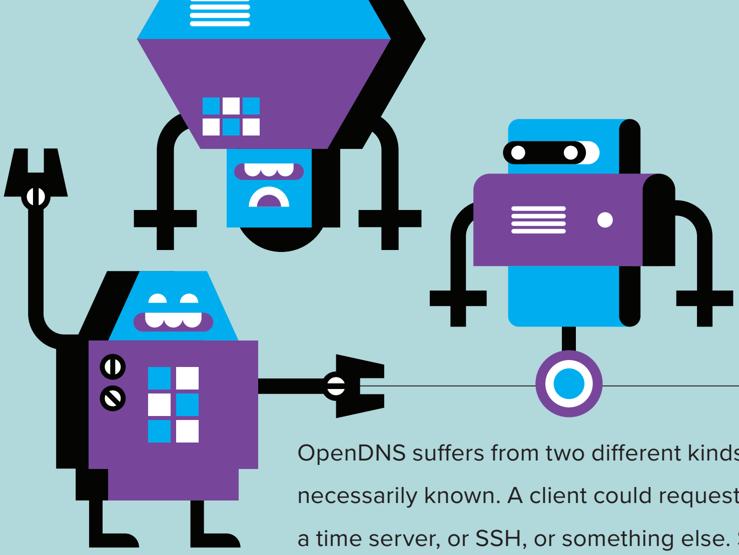
When we spoke with other researchers at a conference, we learned that they had recently decided to switch from Alexa to the OpenDNS Top 1 Million list. OpenDNS runs—you guessed it—DNS servers with added security. So, for example, OpenDNS can detect phishing sites and provide threat intelligence on malware hosting. Many hosts use OpenDNS to the tune of 150 billion queries per day.[ix]

So, we took the two lists, OpenDNS and Alexa, and compared them against each other. Only 17% of hosts appear in both databases. That means that there are nearly two million domains in the union of the lists. For this report, we used both lists (removing the duplicates) as “popular sites” source data. Only a third of the union (approximately 672,000) offer SSL/TLS.

FIGURE 9

17% OF INTERNET HOSTS APPEAR ON BOTH THE ALEXA AND OPENDNS “TOP 1 MILLION SITES” LISTS





OpenDNS suffers from two different kinds of bias. The first is that the “service” a client requests isn’t necessarily known. A client could request a DNS lookup for a non-HTTP service, for example. Maybe a time server, or SSH, or something else. Second, lots of robots use OpenDNS, so it isn’t measuring “human popular sources,” per se.

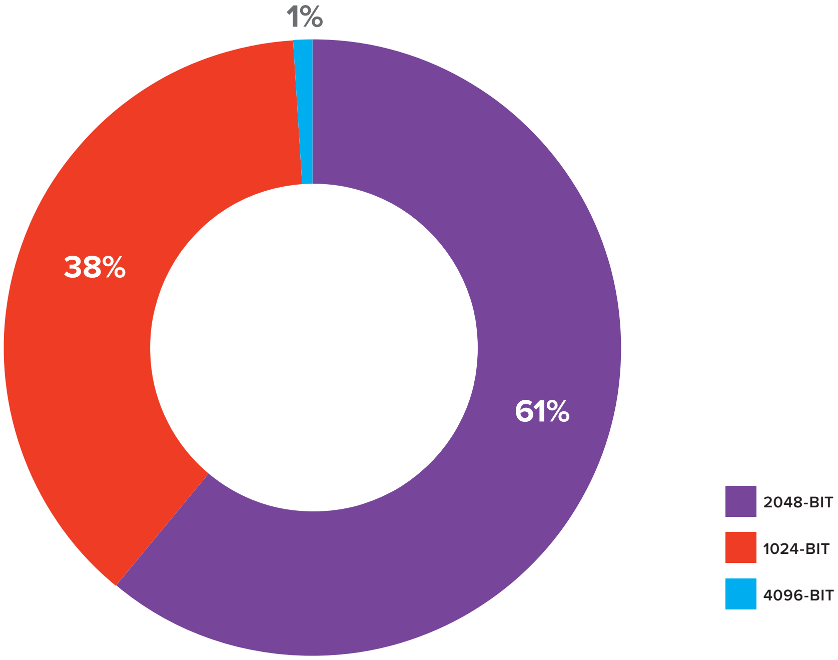
The important thing is that the combination of the two lists gets us close to one million hostnames, and that’s the only way we can see hosts that require a server name indicator, which is at least 2% of the population (see methodology appendix).

KEY SIZE SURPRISE

When we started our research during the summer of 2014, the world was basically split into two camps: those still using 1024-bit RSA keys, and those who had upgraded to 2048-bit keys. About 1% had pre-upgraded to 4096, and about 0.4% were still using 512-bit keys.

FIGURE 10

RSA KEY LENGTHS IN USE IN 2014

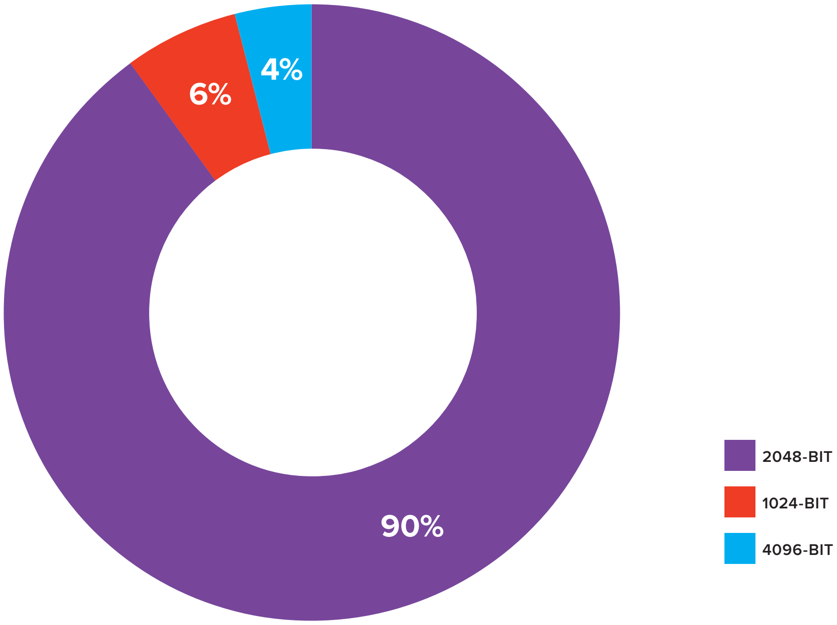


The trend to leave 1024-bit keys behind has been steady. Today, those keys are used by fewer than one in 10 hosts. Yes, it’s totally the same hosts who are using self-signed certificates! The correlation is nearly one to one, confirming our theory that there are many “forgotten” devices out there on the Internet.

Not surprisingly, 2048-bit keys now protect 90% of the world's TLS hosts. The use of 4096-bit keys have quadrupled, as a share of hosts, but we expect it to start shrinking again as Elliptic Curve Digital Signature Algorithm (ECDSA) gains favor.

FIGURE 11

RSA KEY LENGTHS IN USE IN JANUARY 2018



The world is following NIST's Suite B advice. Instead of moving to 4096-bit keys, more and more sites are switching to elliptic curves of a strength equivalent to 3072-bit keys which, according to BlueKrypt, should be good until 2030.[x] Many of those sites are leaving RSA keys behind and using ECDSA keys. The vast majority of which are 256-bit, but there is a scattering of 384- and 512-bit keys in there, as well.

TABLE 2

DSA KEYS USED BY COUNT AND PERCENTAGE

DSA KEY SIZE	COUNT	PERCENT
256	222,426	99.6%
384	821	0.4%
512	14	0.0%

WHO'S STILL USING 512-BIT RSA KEYS?

512-bit RSA keys have been considered unsafe since the mid-1990s. They were successfully factored in 1999, proving their insecurity. Today, you can break a 512-bit key in 4 hours for about $75 on a cloud-based factorization service.[xi] NIST has been recommending 2048-bit keys for nearly a decade.[xii] So, we're kind of surprised to see any 512-bit RSA keys out there. This being the Internet, of course there are some.

TABLE 3

TOP 10 512-BIT RSA KEYS STILL IN USE

RANK	SERVER	COUNT
1	RomPager/4.51 UPnP/1.0	358
2	WindRiver-WebServer/4.7	209
3	GoAhead-Webs	154
4	(unknown)	108
5	Radware-web-server	87
6	Cisco-IOS	50
7	Apache	50
8	lighttpd/1.4.18	45
9	RGOS HTTP-Server/1.1	39
10	WISPR/2.63 WISPR-SSL/2.91	30

In our latest sample of nearly 5 million TLS hosts, after filtering out 512-bit DSA keys, we found only about 1,500 that are still using 512-bit RSA keys. The top ten hosts, as identified by the HTTP server string, are shown in Table 3. Most are ancient embedded TLS stacks, which is about what you'd expect. The number one entry, RomPager, has several associated CVEs—and they could probably be easily hacked, if they haven't been hacked by thingbots already, as we point out in our most recent F5 Labs IoT report, The Hunt for IoT: The Growth and Evolution of Thingbots Ensures Chaos.

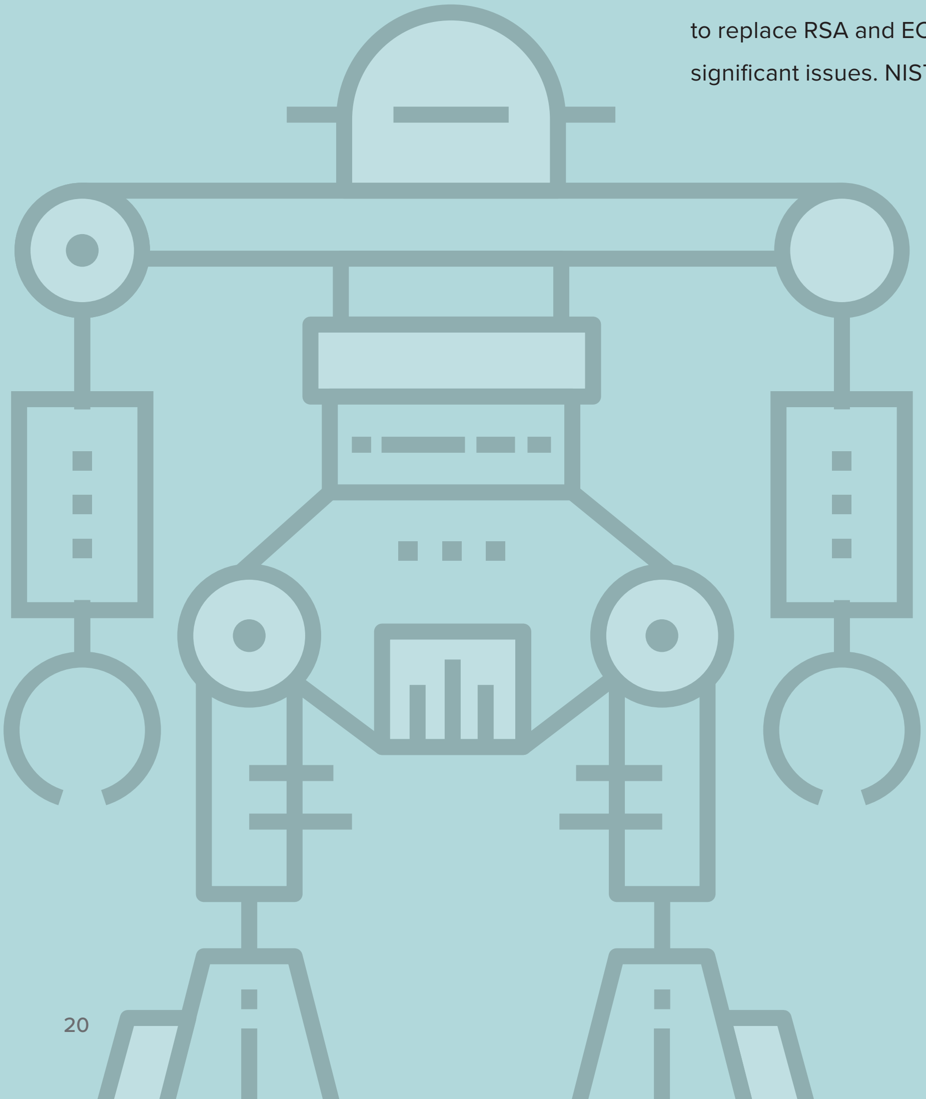
YOU CAN BREAK A 512-BIT KEY IN 4 HOURS FOR ABOUT $75 ON A CLOUD-BASED FACTORIZATION SERVICE.

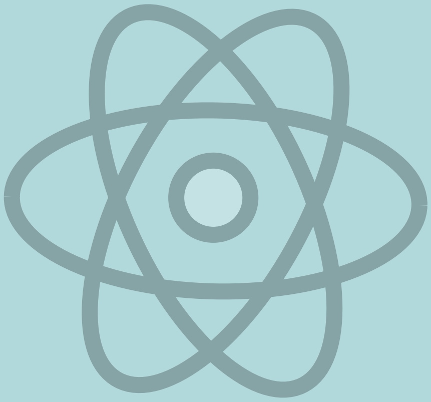
THE THREAT OF QUANTUM COMPUTERS

As part of our 2017 crypto research, we look at the threat of quantum computing as it relates to SSL/TLS. The threat of quantum computers (QCs) is that they may be able to solve previously unsolvable problems through the magic of quantum mechanics with their nifty super-imposed states. A QC of sufficient size—4,000 quantum registers—could crack RSA, ECC, and Diffie-Hellman. Not even the existing forward secrecy ciphers would keep data encryption safe.

Current quantum computers are limited to double-digits of qbits, so they are a long, long way off from being ready for the prime-time key cracking. But if nation-state actors or malicious types really are recording much of the encrypted Internet traffic, they may someday be able to crack today's communications with a QC from the future. Which is why we need to be researching quantum-resistant algorithms now.

There are a handful of quantum-resistant algorithms being considered to replace RSA and ECC key exchanges in TLS, but they all have significant issues. NIST is currently evaluating the applicants—which likely include McEliece with Goppa Codes, Ring Learning with Errors, and Supersingular Isogeny Diffie-Hellman (SIDH)—and will report findings between 2020 and 2022. You can read more about these algorithms and the difficulties of quantum-resistant cryptography in the F5 Labs report, How Quantum Computing Will Change Browser Encryption.





RECOMMENDATIONS

In our first TLS telemetry report that looked the cryptographic trends from 2016, we made five recommendations to help organizations improve the security posture of their SSL stack. Here's our current list of recommended actions for the coming year:

HTTPS EVERYWHERE

Our number one recommendation last year was to stop using self-signed certificates. That's still good advice, and it seems the Internet is taking it to heart, as self-signed certification usage dropped significantly between our two reports. Much of that is due to the open CA, Let's Encrypt.

This year we're making a similar recommendation: it's time for every application to switch to HTTPS. Google is deducting page rank points for insecure HTTP servers and is also marking all HTTP sites as "Not Secure" in their Chrome browser. For all your applications that are HTTP only, it's time to get a Let's Encrypt certificate, spin up a new virtual server on port 443, and change your HTTP virtual server to be a simple redirect. It's easy enough to do, just watch a couple of DevCentral videos for more information.

GET ABOARD THE STRICT TRANSPORT SECURITY TRAIN

While HTTP Strict Transport Security (HSTS) adoption has been climbing, overall adoption remains low. Despite implementation challenges, without HSTS it doesn't really matter how cryptographically secure your website is if a client can be redirected away before it even has an opportunity to see your certificate.

We recommend that all new sites employ HSTS from the beginning. Then, perform an annual gap analysis of non-HSTS. Examine all subdomains—ensure that they're ready for HTTPS. For those that aren't, consider deploying a TLS proxy from an ADC (a quick fix that requires few or no back-end changes).

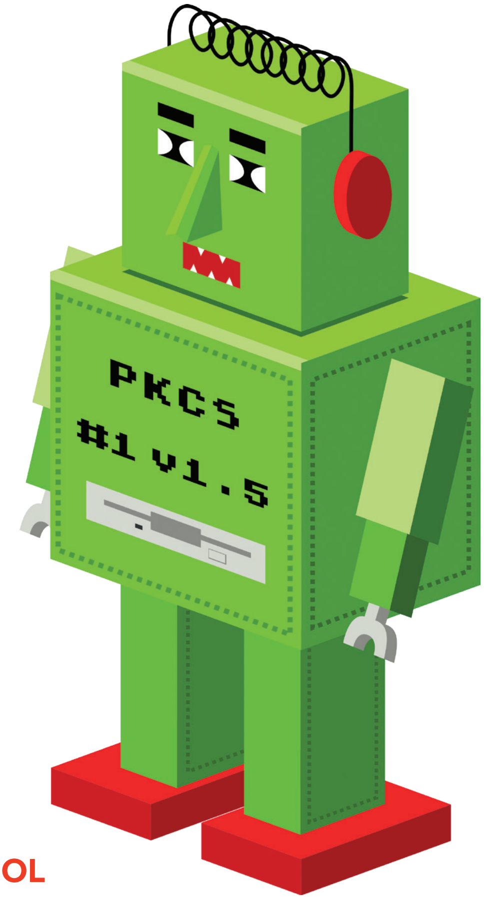
If you think you're ready to make the jump, employ HSTS for a few days in test and production. This will keep you from inadvertently black-holing your site for several months if you actually aren't ready.

Use the Google HSTS portal[xiii] to check your website for HSTS readiness.

TURN ON OCSP STAPLING

Our advice on Online Certificate Status Protocol (OCSP) stapling remains the same as last year. With OCSP stapling, a TLS termination device can insert a signed copy of the certificate status directly into the TLS handshake. This reassures the client that the certificate hasn't been revoked, and it doesn't require a separate connection to a questionable OCSP server.

For sites that aren't using short-lived certificates (that includes every site that isn't using Let's Encrypt), you should definitely turn on OCSP stapling. It's simple to do on an F5 device. SSLMate also has a handy guide[xiv] describing how to enable OCSP stapling for Apache and NGINX, if you happen to be terminating SSL/TLS on those servers instead.



PATCH FOR ROBOT

At the end of 2017, researcher Hanno Bock and team discovered that many SSL/TLS vendors were vulnerable to an ancient attack called the Bleichenbacher, also known as ROBOT. F5 was one of them; of course, we had originally coded to avoid Bleichenbacher attacks, but it slipped in as a regression during a crypto-offload firmware integration.

Most users have upgraded already, but it's worth taking a few minutes to make sure that all your appliance and virtual editions are up to the latest patch levels. We wrote extensively about ROBOT on F5 Labs and DevCentral.

QUALYS SSL SERVER TEST STILL A GREAT TOOL

The Qualys SSL Labs server test remains one of the best and easiest ways to check your site's cryptographic security posture. The test parameters that determine your site's letter grade change over time. Qualys SSL Labs has recently released the grading changes for 2018.[xvi] Significant changes to the grading system include:

- Penalty for not using forward secrecy (B)
- AEAD suites are required to get a A+
- Penalty for the ROBOT vulnerability (F)
- Penalty for using Symantec certificates

And there are other changes, as well. See the Qualys website to prepare your site configuration for the coming year.

CONCLUSION

Cryptography continues to fascinate the paranoid and the propeller-heads—and we say that affectionately because we at F5 Labs are both. Business people know they need cryptography to be “safe” on the Internet. And the rest of the world is just hoping that we, the crypto community, are building protocols to, at the very least, protect confidentiality in the digital world.

In our 2016 TLS Telemetry Report, we measured the basics of cryptographic security posture on a global scale. In this, our 2017 edition, we looked deeper into the cutting-edge topics like elliptic curve choice, TLS 1.3 acceptance, and quantum-resistant cryptography. Until the fabled day comes when everyone is using the same, super-safe, quantum-resistant algorithm, researchers at F5 Labs will continue gathering the statistics showing that some are safer than others, and some have been left behind. You can subscribe to the F5 Labs mailing list for updates to our cryptographic, IoT, and malware research.

APPENDIX

F5 LABS' INTEREST IN SSL/TLS

At F5 Labs, we have an interest in the "encrypted world" not just because a significant percentage of that world is decrypted by F5 devices, but because privacy is a central concern for *everyone* in the fully connected world we live in today. In mid-2014, we began sampling Transport Layer Security (TLS) hosts on the Internet with the goal of collecting and aggregating global metrics for TLS protocol selection, cipher selection, and overall cryptographic security posture. Our TLS telemetry reports chart the trends of cryptographic data in the encrypted world.

OUR SCANNING METHODOLOGY

F5 Labs pulls lists of known TLS hosts from Project Sonar's SSL/TLS known-hosts lists. Between 2014 and 2016, Project Sonar has been tracking approximately 28 million known TLS hosts on the Internet.

Project Sonar's datasets are approximately double that size (60 million lines) because when a host offers multiple certificates from a single IP address, each certificate and IP address pair is counted as a different entity.

We use the list of millions of IP addresses from Project Sonar as a starting point. When the F5 TLS scanner first started in 2014, it sampled a subset of the Internet using lightweight probes that collect the TLS characteristics of the individual sites found. The scanner has been sampling approximately between one and five million TLS sites per quarter.

IP-based Scanning Versus Popular Domains

The IP address-based host data from Project Sonar gives the most complete picture of TLS servers on the Internet. However, a blind TLS request to many IP address servers may fail due to the Server Name Indicator (SNI) problem. That's because SNI multiplexes certificates through a single IP address. For example, suppose that both www.example.com and www.domain.com resolve to the same IP address, 1.2.3.4. When a browser connects to address 1.2.3.4, it can indicate via the SNI extension which site it would like to see—say, www.example.com. The server can then provide the correct certificate for www.example.com and the connection can proceed. If a browser or scanning device connected directly to the address (https://1.2.3.4) without specifying a domain, the result is defined. Some servers will serve a default certificate, but many will refuse the connection. In their paper, *Towards a Complete View of the Certificate Ecosystem*,[xvii] J. A. Halderman, et al. document the number of non-responding SNI servers to be at least 1.5%. For this reason, F5 Labs researchers use both IP-based scans and scans via popular domains, which we pull from both the Alexa and OpenDNS top million lists (see previous section of this report). At times, we discuss statistics from the "overall" population set (mass scans), and other times from the "popular" site list where appropriate.

ENDNOTES

i https://citizenlab.ca/2018/03/bad-traffic-sandvines-packetlogic-devices-deploy-government-spyware-turkey-syria/

ii https://www.securityweek.com/convergence-replacement-throwdown-dane-vs-tack-vs-ct

iii https://devcentral.f5.com/Portals/0/Cache/Pdfs/2807/mutations-in-the-tls-protocol-hsts-false-start-snap-start.pdf

iv https://en.wikipedia.org/wiki/QUIC

v https://blog.cloudflare.com/why-tls-1-3-isnt-in-browsers-yet/

vi https://cr.yp.to/ecdh.html

vii https://csrc.nist.gov/News/2017/Transition-Plans-for-Key-Establishment-Schemes

viii https://en.wikipedia.org/wiki/EdDSA

ix https://system.opendns.com/

x https://www.keylength.com/en/4/

xi https://arstechnica.com/information-technology/2015/10/breaking-512-bit-rsa-with-amazon-ec2-is-a-cinch-so-why-all-the-weak-keys/

xii https://devcentral.f5.com/articles/f5-friday-the-2048-bit-keys-to-the-kingdom

xiii https://hstspreload.org/

xiv https://sslmate.com/blog/post/ocsp_stapling_in_apache_and_nginx

xv https://devcentral.f5.com/articles/return-of-bleichenbacher-the-robot-attack-cve-2017-6168-29119

xvi https://community.qualys.com/docs/DOC-6321-ssl-labs-grading-2018

xvii https://jhalderm.com/pub/papers/https-perspectives-imc16.pdf

APPLICATION THREAT INTELLIGENCE

F5 Labs combines the threat intelligence data we collect with the expertise of our security researchers to provide actionable, global intelligence on current cyber threats—and to identify future trends. We look at everything from threat actors and the nature and source of attacks, to post-attack analysis of significant incidents in order to create a comprehensive view of the threat landscape. From the newest malware variants to zero-day exploits and attack trends, F5 Labs is where you'll find the latest insights from F5's threat intelligence team.

For more information, visit www.f5.com/labs.

US Headquarters: 401 Elliott Ave W, Seattle, WA 98119 | 888-882-4447 // Americas: info@f5.com // Asia-Pacific: apacinfo@f5.com // Europe/Middle East/Africa: emeainfo@f5.com // Japan: f5j-info@f5.com