SANDVINE

Encryption and DPI: Current and Future Services Impact

The growing prevalence of encryption on the Internet is a welcome development for the Internet at large as it dramatically increases consumer confidence in privacy. It also provides an opportunity for service providers and DPI vendors to concentrate on use cases benefiting the user experience rather than ones perceived to be invasive (and controversial).

This paper outlines the current state of affairs in encryption and how it impacts DPI services, and speculation on where encryption on the Internet is heading in the upcoming years - save your copy and mock us if history proves us wrong.

THE CURRENT STATE OF ENCRYPTION ON THE INTERNET

Internet encryption is not a new development - it has been with us for over 20 years[1]. The use of encryption has increased as standards matured and especially as processing power has increased to the point where the CPU impact of encryption is a trivial concern rather than a major one. In the past few years we've reached the point where mainstream apps are almost universally running over encrypted connections when dealing with sensitive data. DPI systems have coped with encryption for many years, as applications have used encryption to increase privacy.

The glaring exception to that statement is the World Wide Web, which until recently has been one of the few Internet mainstays where a lack of encryption has been acceptable to users and content providers alike. We are now seeing a rapid change of this status quo.

Encryption: The Consumer Perspective

Consumers have an ambivalent relation to encryption and privacy. Widespread Internet security issues have a tendency to make it into mainstream news, with Firesheep[2] and Heartbleed[3] as notable examples, not to mention the release of the Snowden documents. One report suggests that nine out of ten respondents have heard about governmental surveillance programs[4] outlined by Edward Snowden two years after the fact. However, user behavior doesn't reflect this. Consumers are still disinclined to adopt security measures that require a change in behavior. A study done after Snowden[5] showed an increase in encryption usage by consumers.

[1] IPSec, SSL (now TLS) and SSH all first emerged in or around 1995.
[2] http://edition.cnn.com/2010/TECH/mobile/11/01/firesheep.wifi.security/
[3] http://edition.cnn.com/2014/04/08/tech/web/heartbleed-openssl/
[4] http://www.pewinternet.org/2015/03/16/americans-privacy-strategies-post-snowden/
[5] https://www.Sandvinenetworks.com/blog/2013/07/24/internet-hide-and-seek%E2%80%A6are-consumers-reacting-privacy-concerns

Projects like Pretty Good Privacy (PGP), which provides end-to-end security and privacy for e-mail, still see anemic adoption rates, 25 years after introduction to the market.

Consumers often ignore security warnings unless they are very carefully designed to encourage the secure behaviour by default. A Google study in 2015 suggested that 70% of Chrome users ignored warnings about insecure connections[1], prompting a major redesign of the way the warning was delivered, only to still see a 38% failure rate.

As an acknowledgement of the challenges in changing user behavior, a more aggressive rollout of security standards that do not hinge on changing user behavior has begun by the leaders in Internet technology. Solutions that rely on built-in and default browser capabilities are becoming the security norm, rather than the exception. More sites now connect via secure protocols than ever, and more browsers are highlighting behavior that could be harmful more prominently.

For example, Google has been vocal about an upcoming change in Chrome's behavior with regards to unencrypted web sites - the browser marking them as explicitly insecure[2]. Recent versions of the developer version of Firefox (44) already mark pages containing password prompts as insecure unless they are served over HTTPS[3].

Encryption: The Content Providers Perspective

These changes by browsers are also driving adoption by more content providers. Doing nothing introduces a substantial risk of a bounce rate in double digits when the browser changes are enacted and your site is flagged as a security risk.

It is not clear exactly **when** the change in browser behavior will happen, but the intent of both Google and Mozilla has been clearly communicated, and is already starting to occur. It is likely that further announcements in this area will be made in 2016.

> If you're developing a new app, you should use HTTPS exclusively. If you have an existing app, you should use HTTPS as much as you can right now, and create a plan for migrating the rest of your app as soon as possible. [...] If your app needs to make a request to an insecure domain, you have to specify this domain in your app's Info.plist file.
>
> **iOS 9 developer library**

CloudFlare reported their page loading more than 50% faster over HTTP/2.

Google is also adding weight to search results for encrypted content and the major app ecosystem gatekeepers for mobile are encouraging the use of encryption throughout[4], so it is in a content provider's interest to move to using encryption by default.

Enabling HTTPS/TLS for a web site is also a prerequisite for enabling HTTP/2. While the encrypted connection setup comes with some overhead, HTTP/2 improves the typical web page load time enough to offset the connection setup losses and yield considerable gains in page load time on top of that. One of the caching CDN providers, CloudFlare®, reported their page loading more than 50% faster over HTTP/2[5].

Given this, even sites with zero interest in the privacy or security benefits of encryption have a clear driver for enabling encryption to reap the gains in page load time.

[1] https://nakedsecurity.sophos.com/2015/02/03/google-redesigns-security-warnings-after-70-of-chrome-users-ignore-them/
[2] http://motherboard.vice.com/read/google-will-soon-shame-all-websites-that-are-unencrypted-chrome-https
[3] https://twitter.com/rlbarnes/status/656554266744586240
[4] http://motherboard.vice.com/read/apple-wants-to-kill-the-unencrypted-internet
[5] https://blog.cloudflare.com/introducing-http2/

Two initiatives in particular have made it easier than ever before for a web site operator to enable encryption for their content: CloudFlare's **Universal SSL** and ISRG's **Let's Encrypt**.

Universal SSL is a cloud based service allowing deployment of TLS (and as a function of CloudFlare's other offerings, HTTP/2) optionally without forcing the site operator to deal with key management. The service, functioning as a reverse proxy, allows legacy HTTP/1.1 based web properties to serve content via HTTP/2 without upgrading their backend infrastructure.

Let's Encrypt, backed by Mozilla, Akamai, Cisco, EFF and others, offer free certificates and a tool chain allowing a low maintenance approach to HTTPS. A serious contender in the free Certificate Authority space, Let's Encrypt is getting integrated into web hosting solutions, such as cPanel and Plesk, two of the industry standard web hosting control panels.

Between them, CloudFlare and Let's Encrypt are providing means for sites of any size to enable TLS without much in terms of domain knowledge or upfront investment.

ENCRYPTION: THE IMPACT ON DPI USE CASES

DPI has evolved to being a strategic technology for network operators, and enables a wide range of use cases. Encryption has always been a factor in use cases for DPI, and the use cases that can be delivered by DPI solutions will evolve as encryption becomes pervasive.

This paper concentrates on DPI in the service provider environment, where the operator doesn't have control over the endpoints. For instance, enterprise-centric DPI-capable firewalls that depend on the client devices having a specific root certificate installed[1] are not covered. Likewise, systems that require a copy of the private key for the server certificate are not covered in this assessment.

We believe that this shedding of the worst possible cases will have a positive impact on the DPI market as a whole.

Encryption: Sandvine's Perspective

DPI is a broad market - ranging from embedded DPI technology to virtual DPI to hardware-based DPI solutions as well as solutions designed for service providers or enterprises. Sandvine offers solutions in all of these form factors, and has a broader view than solutions that are focused on a single vertical or form factor.

One of the clear benefits that a widespread usage of encryption brings is the elimination of some of the use cases that have driven a negative perception of DPI. Use cases that involve session metadata gathering or even URL-based intelligence can no longer gather potentially sensitive user information. It's no longer a case of what is desired by the operator or regualtory bodies - the use cases are simply not possible.

We believe that this elimination of the negative use cases will have a positive impact on the DPI market as a whole. We also believe that the position taken by security researchers - to see any middleware as a possible attacker on the network - is an accurate one. User privacy is better dictated by something under the control of the user - the operating system and browser - than by something under the control of the operator - the middlebox.

Together with the stance of Apple and Google on end-to-end security, encryption obsoletes initiatives that were being considered: the desire to inspect all traffic in Kazakhstan[23], the legal

[1] This also doesn't cover the - thankfully - isolated cases where certificate authorities cooperate with firewall vendors to offer man-in-the-middle capabilities without a separate root certificate install on the clients.
[2] http://www.csoonline.com/article/3012193/cyber-attacks-espionage/in-kazakhstan-internet-backdoors-you.html
[3] https://bugzilla.mozilla.org/show_bug.cgi?id=1232689

status surrounding the access of device data in the US or the desire of the UK to break encryption[4] and a slew of similar cases. With encryption becoming pervasive, DPI can now focus on delivering value to the operator without risk of privacy concerns.

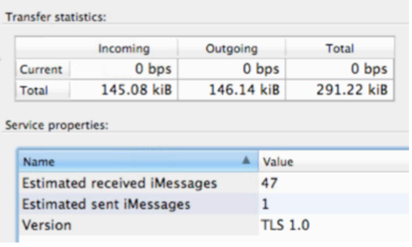
Analytics and Reporting Use Cases

In order to build networks that can meet subscriber expectations, operators need to understand what applications are being consumed on their networks and how the subscriber experience manifests itself. A good experience for Netflix doesn't necessarily indicate that the network supports popular online games well or vice versa. Information is key.

One simple case that's going away is the ability to expose the type of data being transferred in a HTTP session - meaning it is impossible to separate a video file from a software install ISO file - they are both a large number of bytes and streamed to the consumer. It's still possible to infer that it is in fact a download, just not **what** is being downloaded.

More advanced use cases like video analysis that infers a MOS[5] or MOS-like score from a video stream and requires codec and container format awareness and visibility are also dramatically affected. As the data required to determine MOS scores for the algorithm is no longer available, the method of measuring quality for these applications must change.

While a lot of the low hanging metadata is going away, there are still many markers available even in the encrypted flow. The requested name - and for the time being, certificate - of the HTTPS server is still available, giving some insight into **where** a subscriber is connecting, even if the data or type of data being transferred is encrypted.



Transfer statistics:

	Incoming	Outgoing	Total
Current	0 bps	0 bps	0 bps
Total	145.08 kiB	146.14 kiB	291.22 kiB

Service properties:

Name	Value
Estimated received iMessages	47
Estimated sent iMessages	1
Version	TLS 1.0

The flow characteristics can also be leaking information. For example, Procera has been able to count iMessages sent/received by observing the encrypted request by packet size, order analysis, and the encrypted acknowledgement from the server.

> [...] a shift from exposing hard data to inferring knowledge with a degree of uncertainty.

Likewise, the overall behavior of the flow, whether it seems like the endpoints are pushing max MTU packets as fast as they can or if they're presenting a constant flow of small packets in both directions (indicative of interactive two-way media such as VoIP), is also useful for analytics.

This type of analysis of encrypted data isn't new - the iMessage case has been supported since shortly after the service was introduced in 2011 and flow behavior since 2006 - neither is it a unique capability.

The takeaway is that encryption isn't the end of real-time network data analysis, but it does represent a shift from exposing hard data to inferring data with a degree of uncertainty.

From a privacy point of view, this is a good thing for consumers. The market is obsoleting use cases that expose consumer information and cases where an operator was required to get in the middle of a transaction. For instance, the usefulness of video transcoding devices has been superceded by endpoint negotiated capabilities such as the Dynamic Adaptive Streaming over HTTP standard, commonly known as MPEG-DASH and related technologies.

Content caching is also moving closer and closer to the edge, with CDN infrastructure directly peered with or offered by the Internet Service Providers. This enables a far better user experience and resilience against traffic spikes, be they from Denial of Service attacks or from popular content - something not possible with opportunistic HTTP caches.

[4] http://www.nytimes.com/2015/12/22/world/europe/apple-pushes-against-british-talk-of-softening-encryption.html

[5] Mean Opinion Score, originally describing the perceived quality of voice communications, now applicable to a wider range of media. See PEAQ and PEVQ.

“The underlying premises for the acquisition of ByteMobile have now vanished. We acquired the company for its ability to optimize video traffic, but today a significant amount of the video traffic is encrypted and can no longer be optimized”

Citrix, November 2015

Traffic Modification

While actively editing the packets in flight isn’t strictly speaking a function of Deep Packet **Inspection**, DPI is often used in conjunction with use cases such as captive portals, HTTP header injection, video transcoding or even ad insertion, whether it’s performed by the device itself or the device is acting as a Network Packet Broker. These use cases are going away over time, or morphing into ones that require the cooperation of either of the endpoints to function. Captive portals in particular are not quite as easily deployed as they used to be, on account of a greater portion of the traffic being TLS.

This endpoint supported behaviour could for instance be canary URL’s used by browsers and operating systems, allowing for use cases like captive portals for user agreement acceptance, but these use cases will depend on the cooperation of the vendors.
Thus, the power to define acceptable invasiveness is shifting from the Internet provider to the browser vendor[6]. Any solution relying complete visiblity for a major portion of the traffic is going to yield diminishing returns over time as encryption becomes more pervasive.

Major use cases affected are opportunistic HTTP caching and video transcoding solutions, where the entire value proposition hinges on being able to see into traffic and reduce the amount of bytes transferred on the wire. One example of this impact is the recent market exit of **ByteMobile**, due to the marked increase in HTTPS uptake impacting video optimization.

Traffic Management

As opposed to some of the more invasive traffic modification cases, traffic management depends on dropping or queueing traffic rather than rewriting packets. It has been applied to encrypted protocols since the 1990’s - from a technical point of view, the encryption of web traffic offers no change to how traffic management is performed.

Use cases requiring traffic management are mostly unaffected by encryption, unless the criterion used for shaping consists of data that is no longer available. For example, non-interactive services such as software downloads are harder to differentiate from interactive content such as web browsing. In most cases it’s still quite possible to do it, but it requires more advanced techniques or knowledge about the endpoints that isn’t derived from the connection itself.

Quality of Experience information to support the traffic management algorithms is also by and large available even with more widespread encryption.

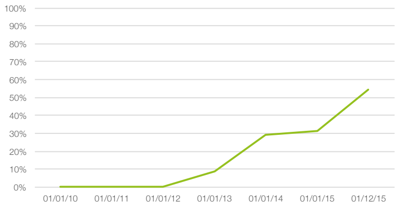
Policy Enforcement

Policy Enforcement is commonly associated with mobile networks and 3GPP Policy and Charging Enforcement (PCEF) functionality with DPI solutions. Many of the use cases for policy are not affected by encryption, but just as described for traffic management, use cases that required access to specific metadata may no longer be possible. Advanced URL filtering that requires full access to the requested URL will not be possible, but site-based URL filtering wil be possible as long as the site name is available for analysis.

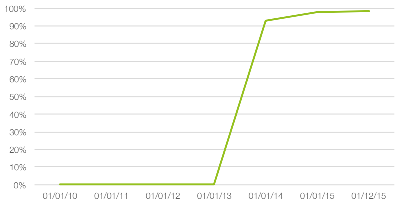
Advanced charging use cases will also be affected - for example cases where users may be charged differently for video streaming or different types of messaging will need to be modified to acknowledge some uncertainty in the collected data.

Zero-rating of applications and content can still be supported, but may require more than traditional DPI - integration with peering/routing technology will increase the assurance that traffic is being rated properly and therefore charged properly.

[6] https://www.chromium.org/chromium-os/chromiumos-design-docs/network-portal-detection



Apache web server install base capable of supporting TLS version 1.2



Google Chrome install base capable of supporting TLS version 1.2

PREDICTIONS ABOUT THE FUTURE

Since Web security has seen a decent amount of change over the past few years, making predictions based on past data and performance is very difficult. But let's try anyway!

The TLS standard has seen incremental updates to counter security issues for it's first 15 or so years in existence, combined with a rather slow uptake of new versions both in coding libraries and in web servers (evergreen[1] browsers made the client uptake a lot zippier than the server side). This has changed in multiple ways over the past few years.

Singular large entities are capable of driving market disruption

When browsers and servers were owned by different entities, development tended to be evolutionary rather than revolutionary. Google, with a substantial footprint both on desktop via Chrome, mobile via Android and as a substantial content provider in its own right, had the unique opportunity to do large scale experimentation across a wide user base and without having to seek support by means of cross-vendor interop proposals.

Experimentation work resulted in the SPDY specification which later formed the basis of HTTP/2 without any major changes. Since HTTP/2 requires TLS for backwards compatibility with HTTP/1.1, this was a unique driver for TLS - enabling it improved the user experience.

Google later went on to experiment with QUIC, a UDP based stack intent on further reducing the latency of secure connection setup by combining the semantics of the TCP and TLS handshake into one singular handshake, even combining it with the ability to optionally send request data on the first packet of a secure connection if the client and server had previously exchanged security information.

Cisco is another vendor that is well poised for large scale disruption, albeit in a niche market: the Internet of Things. IoT presents a new challenge in the number of devices deployed and the lack of traditional interfaces. Owning both OpenDNS (and as such, dnscrypt[2]) and Jasper (an IoT platform vendor) allows them a unique position to push secure name resolution. DNS being one of the last remaining mainstream unencrypted internet protocols, this is an obvious target for attacks or passive analysis.

Google, themselves a popular open DNS provider, is also well poised to close the DNS loophole and enforce encryption between Chrome/Android and their own servers, should they choose to do so. It's conceivable that they could drive future standards by means of large scale experimentation, seeing the approach taken with SPDY and QUIC.

Apple is also in a position to drive this type of disruption on a limited scale if they move in that direction. While they have so far remained content with pushing existing standards rather than being disruptive in quite the same way as Google, they were an early (and to this date only major) adopter of Multipath TCP[3].

[1] Browsers that update automatically without prompting the user to manually download a new version.
[2] Essentially DNS services access via a VPN - https://www.opendns.com/about/innovations/dnscrypt/
[3] iOS: Multipath TCP support in iOS 7 - https://support.apple.com/en-us/HT201373

Post Heartbleed exodus

Platform	Moved to
Amazon cloud	s2n
Android	BoringSSL
Chrome	BoringSSL
Google services	BoringSSL
OpenBSD	LibreSSL
OSX	LibreSSL

Major platforms that has moved away from OpenSSL. Some, like Apple, use multiple stacks whereof OpenSSL used to be one.

The implementation landscape

Actor	Implementation
Amazon	s2n
Apple	Secure Transport
F5	NATIVE
Google	BoringSSL
Microsoft	SChannel
Mozilla	NSS
OpenBSD	LibreSSL
Oracle	JSSE

Covering only major SSL/TLS stack vendors maintaining their own TLS implementations. Many less common stacks exist, although representing less overall network traffic.

TLS stacks are better geared for faster feature turnaround

It was quite common up until 2014 for large vendors and Open Source projects to depend on the OpenSSL library for TLS. The Heartbleed bug[1] changed this landscape by the revelation of OpenSSL being sensitive to the inclusion of security critical bugs with little sanity checking and having a code base that's overall hard to maintain.

This prompted several responses and saw the major vendors move over to different code bases or forking OpenSSL for their own purposes. Many of the companies that define the Internet by means of operating large swathes of clients and/or high bandwith/visitor destinations now maintain their own TLS codebases or use one of the drop-in replacements for OpenSSL.

Notably, Apache and nginx, the two major open source web servers, still rely on OpenSSL.

This new focus and recent investment in plumbing is likey to affect the adoption rate for new TLS versions and extensions, bringing new standards to market a lot faster than previous versions of TLS did, especially when there is a clear business driver behind it.

TLS 1.2 - RFC 5246

Implementation	Days to ship
NSS	1781
OpenSSL	1307
SChannel	341

TLS extensions (SNI) - RFC 3546

Implementation	Days to ship
NSS	1050
OpenSSL	2028
SChannel	1237

The above tables list the implementation inertia in days, measured from the publication date of the relevant RFC. It doesn't tell the whole story, as web servers and clients do use prerelease versions or backport a specific new feature before the official release. Additionally, the TLS extensions RFC underwent several modifications (RFC 4366 & RFC 6066) that prompted implementors to delay. It does, however, highlight the historical inertia.

The next version of TLS has some meaningful business drivers attached to it. While it's hard to speculate in exactly when it'll be implemented by whom, it is likely a safe assumption that the lead time from spec to implementation is going to be shorter than for previous versions.

Security > Performance > Privacy

It seems likely that the main driver for future web cryptography work will be mostly driven by security and performance considerations over privacy considerations. We are at a point, standards wise, where the major bottlenecks in connection setup are either addressed or being addressed, with a focus on improving performance.

Adding privacy enabling features on top of this is likely to decrease, rather than increase, performance and is likewise likely to increase the security risk introduced by complexity. Even if the entire connection is encypted, the endpoints are still known and information can be inferred from that, yielding diminishing value in protocol privacy work.

To get a high degree of privacy, users will have to resort to Tor, Freenet or some similar onion routed protocol with multiple layers of encryption - and even so be aware of the security implications driven by the applications they run when connecting back to the open Internet[2]. It does also come with performance tradeoffs.

[1] http://heartbleed.com
[2] https://www.torproject.org/download/download-easy.html.en#warning

Increased standards convergence is a likely development

Over the past ten years we have observed a high degree of convergence from custom legacy protocols to HTTP as a rough, but not necessarily perfect, fit for many different use cases. For instance, anything involving file transfers is more likely than not to be HTTP/HTTPS based. One such example would be Apple's messaging services where text messages are carried in a wire protocol whereas media files are transferred over HTTPS in a separate connection. Many services also expose API's over HTTP rather than over some specialized and more efficient transport. There are many plausible reasons for this:

- HTTP is a well understood protocol by most developers.
- HTTP libraries tend to exist as standard libraries for any non-obscure programming environment. The development effort required is minimal.
- Networking equipment, including middleboxes, are likely to understand HTTP.

HTTP doesn't come without drawbacks, however. HTTP/1.1 sports head of line blocking - requests are typically not sent in parallel (and if they are, they have to return in-order rather than in whatever order they finish) meaning that 20 requests for small content running over the same connection would incur 20 round trips between the client and the server[1].

HTTP/2 deals with much of the historical HTTP baggage and supports multiplexing and server initiated push. While HTTP/2 capable libraries are by and large still using HTTP/1 protocol semantics and benefiting from the performance improvements, it's conceivable that the next generation of libraries will be closer to HTTP/2's native protocol semantics and allowing for richer expression by the developer.

It would, for instance, be quite possible to produce a client-server based instant messaging client capable of sending voice, video, text and files, all over the same HTTP/2 connection, benefiting from the maturity of standard libraries.

QUIC is also likely to improve this situation once standardized. Beyond the most latency sensitive applications, it is conceivable that we'll continue seeing an even greater congregation towards HTTP/2 or QUIC.

The Certificate Authority situation

The entire TLS security model, at least in the context of the public Internet, assumes that the client can use a trusted third party to verify the identity of the server. This list of entities trusted with the right to issue certificates includes dozens of private entities, a number of national governments and even some regional governments such as that of Valencia, Spain or agencies such as Försäkringskassan, the Swedish social insurance agency.

The actual number of entities that can issue certificates is even greater. In order to power initiatives such as Universal SSL by CloudFlare, trust is delegated from Comodo - the CA - to CloudFlare that the domain in question is in fact under the control of the CloudFlare user.

There has been some high profile cases involving unauthorized certificates being issued fraudulently[2], by negligence[3] or even willingly[4][5].

It's conceivable that the current Public Key Infrastructure CA model will come under even greater scrutiny than it already has and that vendors big enough to be disruptive might drive change, should the situation deteriorate further.

Recognized root CA orgs per vendor

Apple	81
Debian	82
Mozilla	61
Microsoft	115

Numbers represent distinct organizational entities rather than the actual number of root certificates. Not counting organizations representing CA's unable to authenticate servers.

Due to the non-obvious nature of ownership - where one organization may be fully owned by another - numbers are approximate. Likewise, variations per platform or OS version may occur within the purview of each vendor.

[1] Akamai has a demo of this behaviour, comparing HTTP/1.1 vs HTTP/2 - https://http2.akamai.com/demo
[2] https://en.wikipedia.org/wiki/DigiNotar
[3] http://www.symantec.com/connect/blogs/tough-day-leaders
[4] http://www.crypto.com/blog/spycerts/
[5] https://www.trustwave.com/Resources/SpiderLabs-Blog/Clarifying-The-Trustwave-CA-Policy-Update/

TLS working group charter highlights:

- Develop a mode that encrypts as much of the handshake as is possible to reduce the amount of observable data to both passive and active attackers.
- Develop modes to reduce handshake latency […] aiming for one roundtrip for a full handshake and one or zero roundtrip for repeated handshakes.
- Update record payload protection cryptographic mechanisms and algorithms to address known weaknesses [...]
- The WG will consider the privacy implications of TLS1.3 and where possible [...] aim to make TLS1.3 more privacy-friendly
- [...] the TLS WG will also place a priority in minimizing gratuitous changes to TLS.

https://datatracker.ietf.org/wg/tls/charter/

THVE NEXT TLS VERSION, 1.3/2.0

TLS version 1.3 - which for all practical purposes could be seen as a version 2.0 of the protocol - is the next major iteration. It's a security rehash, doing away with cryptographically insecure ciphers, as much as a vehicle for standards cleanup and feature development.

While the standard is at the time of writing not finalized, with a last call date informally announced for Q1 2016, the major themes have been fleshed out and echo the stated goals of the working group (see bullets to the left). That said, this entire section is describing a likely but not guaranteed outcome.

Relation to other encryption standards

The TLS working group has direct or indirect influence over other encryption standards. As a considerable amount of work has been poured into reviewing TLS 1.3 both from a cryptography perspective[1] and from an engineering perspective, capitalizing this makes a lot of sense for other projects.

One such sibling protocol is Datagram Transport Layer Security, DLTS for short, a version of TLS that uses UDP rather than TCP as a transport mechanism. Commonly used for secure WebRTC traffic and for some VPN solutions, notably AnyConnect from Cisco and Edge VPN from f5.

Another one is QUIC, as Google has announced that QUIC's proprietary handshake will be replaced by the standard mechanism in TLS 1.3 once this has been published. The TLS 1.3 handshake mechanism was itself inspired by the one in QUIC.

Privacy changes affecting DPI

TLS 1.3 encrypts the handshake to a greater extent than TLS 1.2. Specifically, the server certificate is transferred in an encrypted format rather than in plain text. It will be impossible for a DPI device to verify the authenticity of the server based on the certificate chain and the known trusted Certificate Authority roots. The Server Name Indication (SNI) extension required for TLS/HTTPS virtual hosting is mandated and still transferred in the clear for a full handshake.

While encrypted SNI has been a topic of discussion, there is no known technical means of encrypting SNI without sacrificing performance.

All encrypted packets are to be sent with the record type of Application Data, where the record type used to be a function of whether the data being transferred was control or application data. This makes content analysis harder.

The specification also defines content padding, allowing blank data to be appended to the payload before encryption. This does expand the size of the payload, expending additional bandwidth, but yields the benefit of making content analysis harder. To what extent content padding will be used in actual implementations remains to be seen.

As mentioned previously in the use case explorations, this could have an effect on use cases like parental control that require specific hostname visibility.

[1] miTLS: A Verified Reference Implementation of TLS - http://www.mitls.org

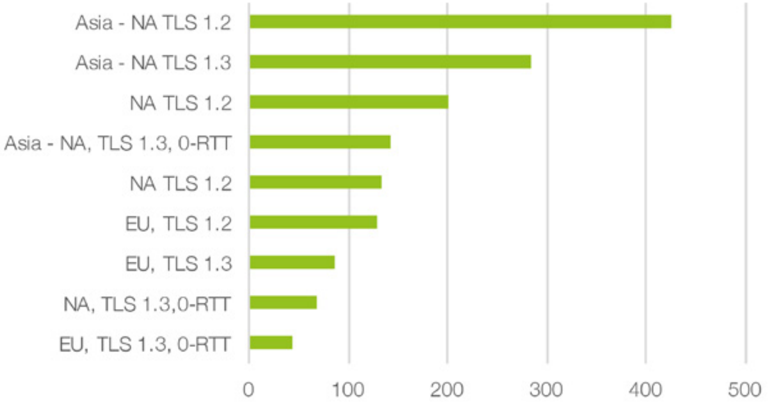
Connection setup performance: 2-RTT to 1-RTT to 0-RTT

The current TLS (1.2) connection setup requires at least two round trips worth of handshakes between the client and the server before actual any application data can be exchanged.

FCC, the US telecoms regulator, reports that the 2015 average terrestrial access network round trip time during peak was 31.38 ms[1]. This does not include latency incurred across the public Internet, which would be limited by the speed of light. One provider, Verizon, reports 11.5 ms average round trip time within Europe, 35.5 ms within North America and 110.09 ms for trans Pacific round trips[2].

Consequently, the real world experience gain from limiting the number of round trips, especially for connections spanning continents, is substantial[3].

TCP + TLS connection setup time in milliseconds



The TLS 1.3 specification defines a handshake requiring one round trip - 1-RTT - for a full handshake and potentially zero round trips - 0-RTT - for subsequent connections to the same server. Caveats apply: it remains to be seen how often the 0-RTT mode will be used and whether there will be any security issues stemming from improper implementations of it, as it potentially allows limited replay attacks to occur.

Nevertheless, going from 2-RTT to 1-RTT is a meaningful improvement. It is also likely that web browsers will implement 0-RTT for idempotent requests such as HTTP GET/HEAD, but fall back to 1-RTT for any other type.

For the technically minded, there is a very good writeup of the changes in handshake behavior between TLS 1.2 and TLS 1.3 in a post by Tim Taubert (Mozilla) titled **More privacy, less latency**[4].

[1] https://www.fcc.gov/reports-research/reports/measuring-broadband-america/charts-measuring-broadband-america-2015#chart7
[2] Measured within the network core - http://www.verizonenterprise.com/about/network/latency/
[3] https://www.fastly.com/blog/thoughts-why-speed-matters-fastly-hits-1-tbsecond-bandwidth
[4] https://timtaubert.de/blog/2015/11/more-privacy-less-latency-improved-handshakes-in-tls-13/

Encryption and DPI: Summary of Service Impact

A more widespread use of web encryption is a market disruptor in many ways. It will force entire segments of networking equipment vendors to either radically adapt or exit the market. It will also force many use cases that have been viewed as negative by Internet users to become obsolete and technically impossible. This is a good thing for the Internet and its users.

Indications suggest there will be a clear increase in web encryption over the next few years, for reasons of both performance and privacy. It will eventually reach a pivot point where it is safe enough for browser vendors to mark unencrypted connections as explicitly insecure, which will be a very large driver for change for the remaining providers of content without encryption.

However, this does not mean the end of visibility into user traffic. It is still possible to see behaviour, visited destinations and similar data even when web encryption is widespread. Granularity and accuracy will suffer, requiring DPI equipment vendors to adopt to this reality and focus on QoE-enhancing use cases. Network operstors that have a good understanding of how their subscribers consume bandwidth and how their network is delivering QoE will have a competitive advantage.

Many cases related to modifying traffic will yield diminishing returns as more and more web traffic goes encrypted. We have already seen market exits and more will come.

There are still many security challenges to tackle for the Internet community. The Certificate Authorities, powering the Public Key Infrastructure that is powering TLS, is a diverse collection of entities in many jurisdictions. There are still critical protocols that are by and large unencrypted or susceptible to tampering, such as BGP or DNS.

There are also large singular entities, such as Google, that are more than capable of driving disruptive change in networking. This has proven to beneficial to service providers and end users alike.

It also means that the future is difficult to predict. We live in an era where technology has moved faster than regulations; where patterns of interaction has changed more drastically in mere decades than in the preceding centuries. Making sense of the Internet world is hard, but it is clear that there are great benefits to be reaped by the ones who manage to do so and those bold enough to set out on new paths.

May we live in interesting times.

v20171219

ABOUT SANDVINE

Sandvine helps organizations run world-class networks with Active Network Intelligence, leveraging machine learning analytics and closed-loop automation to identify and adapt to network behavior in real-time. With Sandvine, organizations have the power of a highly automated platform from a single vendor that delivers a deep understanding of their network data to drive faster, better decisions. For more information, visit **sandvine.com** or follow Sandvine on Twitter at **@Sandvine.**

SANDVINE

USA
47448 Fremont Blvd,
Fremont,
CA 94538,
USA
T. +1 510.230.2777

EUROPE
Birger Svenssons Väg
28D
432 40 Varberg,
Sweden
T. +46 (0)340.48 38 00

CANADA
408 Albert Street,
Waterloo,
Ontario N2L 3V3,
Canada
T. +1 (0)519.880.2600

ASIA
Ardash Palm Retreat,
Bellandur, Bangalore,
Karnataka 560103,
India
T. +91 80677.43333