Towards Side Channel Analysis of Datagram Tor vs Current Tor

Nick Mathewson and Mike Perry
{nickm,mikeperry}@torproject.org

Tor Tech Report 2018-11-002
November 27, 2018

1 Disclaimers

This whitepaper assumes that you know how Tor works.

There are probably some very good references here that we didn't remember to cite.

2 Introduction

Tor's current design requires that its data cells be transmitted from one end of a circuit to the other using a reliable, in-order delivery mechanism. To meet this requirement, Tor relays need to buffer cells–spending resources, hurting performance, and risking susceptibility to out-of-memory attacks.

In order to improve Tor's performance and resilience, researchers have made several proposals for ways to relax the requirement for reliable in-order delivery. In general, these "datagram-based" proposals would allow relays to drop or reorder cells as needed, and move the responsibility for providing a reliable stream protocol to the endpoints (the client and the exit relays).

But by increasing flexibility for the relays, and by increasing the complexity of the endpoints, these datagram proposals also create some new attack vectors. Before we can deploy any of these designs, we need to consider whether these attacks weaken Tor's security, or whether they are irrelevant given other, stronger attacks against Onion Routing.

This whitepaper tries to list these attacks, and to provide a framework for thinking about them as we move forward with our design analysis.

We hope that this whitepaper will help researchers and others in the Tor community to understand these issues, so that we can work together to find new ideas to analyze and mitigate the attacks described here, and to help deploy a faster and more reliable network while still maintaining our current (or better) security guarantees. We hope that our description of the problem space will inspire, not discourage, future experiments in this area, and help with a holistic understanding of the risks, rewards, and future areas of work.

2.1 A toy system

We will be analyzing a system that differs from Tor in the following ways.

- The link between a client and its guard, and between each pair of relays uses DTLS over UDP: packets can be dropped or re-ordered by an attacker on the link, but not modified, read, or forged. Each DTLS packet contains an integer number of cells.
- Each circuit between a client and an exit traverse several relays, as before. The cells on a circuit are no longer guaranteed to arrive reliably, but can be dropped or re-ordered on the wire, or by a relay.
- To provide reliable service end-to-end, the client and the exit each use a TCP-like protocol to track which application bytes have been sent and received. Received data is acknowledged; dropped data is retransmitted.
- The cryptography to be used for circuit encryption is not specified here.
- A reliable signaling mechanism between relays (to create, destroy, and maintain circuits) is not specified here.

(It is likely that many readers will be able to design a system that resists the attacks below better than the design above. But please remember as you do, that a design which improves a system in one way may constrain it in others, or may offer insufficient benefits to be clearly superior to Tor as it is today. Before we can deploy, we will need not just defenses, but also a systemic way to compare the effect of these defenses, used together, to the Tor status quo.)

3 Some preexisting attacks to consider

To put the datagram-based attacks into context, we'll start out by listing some attacks against the current non-datagram Tor design (and proposed defenses for those, where they exist).

We assume, as usual, an adversary who controls some but not all relays, and some but not all ISPs.

A note on attack power: the accuracy of many of these attacks, particularly the passive ones, depends on the type of traffic being sent, the quantity of similar traffic elsewhere on the Tor network, the quantity of concurrent activity by the same client, the adversary's observation position and data retention resolution, the quantity of padding, and the tendency of the network to preserve or alter packet timing information in transit.

In many cases, we don't have good metrics or evaluation methodology to determine how much harder or easier one attack is than another.

3.1 End-to-end passive traffic correlation attacks.

Here's the gold-standard base-line attack: an attacker who can watch any two points on the same circuit is assumed to be able to realize, without having observed very much traffic at all,

that the two points are indeed on the same circuit by correlating the timing and volume of data sent at those two points.

When one of these points is also linked to the client, and one is linked to the client's activity, this attack deanonymizes the client.

Tor's current design focuses on minimizing this probability, and also shifting its characteristics, through things like network diversity and long-term entry points. The attack may also become harder (and/or slower) when there is a lot of similar concurrent traffic on the Tor network, which means that adding users who use Tor for many things is in itself a form of mitigation.

Proposed defenses in this area include deliberate obfuscation of message volume through padding, and of message timing through random delays, as well as things like traffic splitting and more complex traffic scheduling for loud flows. While we have completed some work on link padding, and are progressing on a deployment for circuit padding, it is not yet clear if we can use these defenses in an affordable way against a correlation attack, and it is hard to measure their effectiveness on a realistic Tor-sized network.

3.2 Data tagging side-channels by relays

If two relays are on the same circuit, they can surreptitiously communicate with one another transforming the data in the RELAY cells, and un-transforming the data before passing it on. Since Tor's current encryption protocol is malleable, this allows them to send a large number of bits per cell.

This attack can also be used when two relays do not know if they are on the same circuit. One relay modifies a cell, and the other one looks for such modifications. If the data is processed by an honest relay, it will destroy the circuit, but the client may or may not notice that the circuit has destroyed. (And the dishonest relay may delay informing the client!)

To defend against this, we plan to replace our encryption with a non-malleable algorithm. See for example proposals 202, 261, and 295.

3.3 Destructive side-channels (internal)

Even if we remove the malleability in Tor's encryption, a smaller side-channel remains: A dishonest relay can destroy a circuit at any time, either by corrupting the circuit or simply sending a DESTROY cell along it. A third party can destroy a large number of circuits at once by remotely attacking a client or relay – either disabling that relay, or making it close circuits because of the OOM handler. (See the Sniper Attack paper.)

If a circuit is corrupted (as would happen if a relay attempted data tagging against one of the non-malleable cryptographic algorithms mentioned above), other points on the circuit can tell which cell is the first corrupted cell. If a circuit is destroyed at one point, other points on the circuit can tell how many cells were sent before the destruction.

It is likely that based on data or traffic patterns, most parties on a circuit will be able to distinguish a prematurely destroyed circuit from one that was shut down normally.

In each case, this attack can be used to send (log n) bits of information per circuit, at the cost of destroying the circuit, where n is the number of cells that might be sent over the circuit in total. Some noise will exist, since we expect some circuits to be prematurely closed on their own. We don't know how much noise.

We also have various heuristics that can attempt to detect if this happens too often; however at best they likely reduce the rate that information that can be sent in this way rather than eliminate it. We also lack methodology to measure the rate of information in this case, to help determine if we can successfully reduce it further.

3.4 Destructive network probes (external)

Though TLS is resilient against many forms of active attacks, it can't resist an attacker who focuses against the underlying TCP layer. Such an attacker can, by forging TCP resets, cause all the entire TLS connection to be dropped, thereby closing all the circuits on it. This kind of attack can be observed at other points on the network in a way similar to the destructive side-channels noted above.

This class of attack seems to be easier against Tor's current design than it would be against (some) datagram-based designs, since datagram-based designs are resilient to more kinds of traffic interference.

3.5 Timing-based watermarking attacks

Hostile relays can also introduce a side channels to a circuit by introducing patterned delays into the cells. For example, a relay could buffer a large number of cells, then transmit a "1" bit by sending a cell in a given time period, or a "0" by not sending cells in that time period.

An attacker can also mount this attack without controlling relays: if the attacker performs a DoS attack against a relay or its traffic, it can observe changes in the traffic volume elsewhere on the network.[1]

The bandwidth of this side-channel will be limited, since other relays on the network will naturally buffer and delay traffic, obscuring the pattern some. There are also limits to how long packets can be delayed before the relay is no longer usable.[2]

Proposals for resisting this type of watermarking attack are mostly of the same type that would be needed for resisting end-to-end correlation. An adversary that can perform active attacks to introduce their own unique traffic patterns intuitively seems much stronger than one that must passively use potentially common patterns. We lack a unified framework to tell us how much stronger this adversary is than the passive one, especially against various defenses.

3.6 Traffic injection attacks

Related to the active timing attack, in some positions (like exit and RP) relays can inject cells that are ignored by the other endpoint. These injected patterns will not impact the user's experience, but will allow unique traffic patterns to be sent and detected by the adversary at crucial times.[3]

[1] See https://www.freehaven.net/anonbib/cache/ccs07-latency-leak.pdf and http://cybercentre.cs.ttu.ee/wp/wp-content/uploads/2017/01/crw2017_final.pdf.

[2] See: Rainbow (https://www.freehaven.net/anonbib/cache/ndss09-rainbow.pdf); Swirl: (https://www.freehaven.net/anonbib/cache/ndss11-swirl.pdf); Backlit (detection): (https://www.freehaven.net/anonbib/cache/acsac11-backlit.pdf)

[3] See https://petsymposium.org/2018/files/papers/issue2/popets-2018-0011.pdf

These injection attacks arise from former adherence to Postel's Maxim. Tor has since departed from this maxim, and instead opted for stricter forward compatibility through feature versioning, but removing instances in the codebase where injected cells can be permitted has proven challenging.

4 Attacks unique to datagram designs

Here are some attacks that are enabled by (or at any rate behave differently under) datagram-based designs.

4.1 Traffic-stream tagging (by relays and internet links)

Because the new system permits a number of transformations on traffic that were not previously allowed, we need to look at how those transformations can be used to attack users.

As a trivial example, any router can relay an arbitrary subset of the cells that it receives on a circuit, in an arbitrary order, due to the exact properties the reliable transport aims to provide. The pattern induced in this way will be detectable by the exit relay when it attempts to reconstruct the stream. Because we explicitly allow this kind of transformation, the circuit will not be killed after a single dropped cell, but rather will continue working silently.

Moreover, any ISP can mount the same attack by dropping and/or re-ordering DTLS calls.

A remote attacker may also be able to mount this attack by flooding any router between a client and its guard, thereby causing some of the DTLS messages to get dropped.

If we are using TCP between client and exit, the acknowledgments sent by each endpoint will provide confirmation about which data it received and which it did not. If instead of TCP, we use some other protocol where the end-points communicate even more information about which packets they did and did not receive, this can provide an even higher-bandwidth side-channel.

The bandwidth of this side-channel is fairly high, since it allows the attacker to send over a bit per cell. But it will be somewhat noisy, since some cells will dropped and reordered naturally.

Padding, traffic splitting, and concurrent activity will increase the noise of this attack; we lack metrics to tell us how much, and we have no framework as of yet to measure the throughput of the resulting side channel in these conditions.

4.2 Traffic Fingerprinting of TCP-like systems

Today, because Tor terminates TCP at the guard node, there is limited ability for the exit node to fingerprint client TCP behavior (aside from perhaps measuring some effects on traffic volume, but those are not likely preserved across the Tor network).

However, when using a TCP-like system for end-to-end congestion control, flow control, and reliability, the exit relay will be able to make inferences about client implementation and conditions based on its behavior.

Different implementations of TCP-like systems behave differently. Either party on a stream can observe the packets as they arrive to notice cells from an unusual implementation. They can probe the other side of the stream, nmap-style, to see how it responds to various inputs.

If two TCP-like implementations differ in their retransmit or timeout behavior, an attacker can use this to distinguish them by carefully chosen patterns of dropped traffic. Such an attacker does not even need to be a relay, if it can cause DTLS packets between relays to be dropped or reordered.

This class of attacks is solvable, especially if the exact same TCP-like implementation is used by all clients, but it also requires careful consideration and additional constraints to be placed on the TCP stack(s) in use that are not usually considered by TCP implementations – particularly to ensure that they do not depend on OS-specific features or try to learn things about their environment over time, across different connections.

4.3 Retransmit-based watermarking

Even if all TCP-like implementations are identical, they will retransmit with different timing and volume based on which cells have been acked or not acked. These differences may be observable from many points on the circuit, or from outside the network. Such retransmissions can be induced from outside the network, by hostile relays, or even by a hostile endpoint that pretends not to have received some of the packets.

We again lack metrics to indicate that it is substantially worse (or not worse) than other similar attacks. Intuitively, the key difference in degree would come from how much easier it is to perform this attack than the delay based watermarking attacks on traditional Tor above.

4.4 Congestion and flow control interference

To the extent that the TCP-like stack uses information learned from one stream to alter its behavior on another stream, an attacker can exploit this interference between streams make all of the streams from a given party more linkable.

All implementations will have some amount of interference, to the extent that their bandwidth is limited. But some may have more than necessary.

4.5 Non-malleable encryption designs only currently exist for in-order transports (or the return of data tagging attacks)

Our proposed defenses against data tagging require us to move to non-malleable encryption, with each cell's encryption tweaked by a function of all previous cells, so that if even a single cell is modified, not only is that cell corrupted, but no subsequent cell can be decrypted.

It seems nontrivial to achieve this property with datagram based designs, since we require that cells on a circuit can be decrypted even when previous cells have not arrived. We can achieve data-based non-malleability by using a per-hop MAC for each cell – but we would no longer be able to get the property that a since altered cell would make the whole circuit unrecoverable. This would enable a one-bit-per-cell side-channel, similar but possibly more powerful than the packet dropping side-channel above. (Because the congestion window is essentially a bit vector of received cells, the adversary in this scenario gets to corrupt cells in carefully chosen ways instead of merely dropping them.)

Perhaps other cryptographic schemes could be found to resist data-tagging in a datagram-based environment or limit its impact, but we'll need to figure out what the requirements and models are.

As a proof-by-example of a mitigating system: Proposal 253 describes a way to send a rolling MAC out of band, to ensure integrity of packets between those cells. But can we do better? Can middle nodes enforce integrity in some other way?

4.6 The risks of success: lower latency strengthens timing attacks?

There are two factors that make timing-correlation and timing-watermark attacks more difficult in practice: similarity between different users' traffic, and distortion in timing patterns caused by variance in cell latency on the network. To whatever extent we successfully reduce this distortion by lowering latency, it seems that we'll make these attacks more powerful.

In particular, geolocation attacks based on observed circuit setup times may get worse.[4]

We're already making improvements to Tor that may make these attacks worse – Tor latency has dropped and will continue to drop due to improvements like KIST, more relays, and better load balancing. Further incremental improvements like explicit congestion control on the existing Tor network will reduce latency even further.

It may be that a more performant Tor becomes less safe than a slower, less usable Tor. On the other hand, a more usable Tor will likely be used by more people, which we know makes many forms of traffic analysis harder (slower?) in general. However, we have no way to measure this tradeoff on many different attack types.

Delay and latency can also be added back in, and this has been a common defense against both active adversaries and timing attacks in the anonymity literature, but such delays have user-facing consequences, unless they are carefully restricted to the cases where the adversary can directly measure RTT and can be amortized away by things like pre-emptive circuit building. In this and other cases, it is also not clear to what degree adding delay is more useful than adding more padding.

5 Towards comparing attacks

A high-bandwidth attack is worse than a low-bandwidth attack. One bit is enough to send "is this the targeted user?", but 32 bits is enough to send a whole IP address.

The impact of these attacks become worse if they can be repeated over time.

An attack that can be performed by an ISP relaying traffic is worse than one that can be performed by a relay. An attack that can be performed remotely against either of these is worse still.

We need some kind of methodology to help us compare the new side channels that datagram transports may enable to the existing side channels in Tor, particularly delay-based and congestion-based side channels. Ideally, these metrics or evaluation methodology would also allow us to compare these side channels under various forms of defense, such as padding.

[4] See again https://www.freehaven.net/anonbib/cache/ccs07-latency-leak.pdf

At the very least, we need some way to compare the side channels in datagram transports to those that already exist.

We also likely need a common reference research prototype and/or platform to experiment with and study, so that attacks and defenses are reproducibly comparable. Reproducibility in attack and defense literature is often not reliable, due to differing implementations, in addition to differing methodology and evaluation frameworks.

6 Open Questions

Why permit reordering? There are schemes (like order-preserving encryption) that we could deploy on middle nodes to prevent reordering, without allowing earlier nodes to differentiate padding from non-padding. Do we derive any benefit by allowing a relay to send cells on a single circuit in a different order than the order in which it receives those cells on that circuit? This may be an answered question in congestion control research, but we lack the domain expertise to know what this tradeoff is.

Related: what cryptography to use? Our current stateful encryption schemes benefit from having access to "all previous cells" when encrypting or decrypting each following cell. If we allow a cell to be {de,en}crypted before previous cells are received, we'll need a new model for onion-routing cryptography – possibly one with significantly bigger headers.

7 Future work

We hope to investigate these issues with researchers and others in the Tor community as we work towards solutions to help scale and strengthen the Tor network. Understanding the risks and rewards that datagram-based transports introduce to Tor is important to help us select designs that both help improve performance but also guarantee safety for Tor users. We hope that by cataloging these risks, future conversations about improved network designs can bring answers and broader improvements. We look forward to working with others interested in helping solve these problems to design a better Tor.

8 Acknowledgments

Our thanks to Chelsea Komlo for many helpful suggestions and comments on earlier drafts of this whitepaper, and for writing the request for future work.

9 Further reading

- Steven Murdoch, "Comparison of Tor Datagram Designs", 2011. https://murdoch.is/papers/tor11datagramcomparison.pdf

- Mashael AlSabah and Ian Goldberg. "PCTCP: per-circuit TCP-over-IPsec transport for anonymous communication overlay networks", 2013. http://cacr.uwaterloo.ca/techreports/2013/cacr2013-09.pdf
- Michael F. Nowlan, David Wolinsky, and Bryan Ford. "Reducing Latency in Tor Circuits with Unordered Delivery", 2013. https://www.usenix.org/system/files/conference/foci13/foci13-nowlan.pdf
- Rob Jansen, Florian Tschorsch, Aaron Johnson, and Björn Scheuermann The Sniper attack: Anonymously Deanonymizing and Disabling the Tor Network", 2013 https://www.nrl.navy.mil/itd/chacs/sites/edit-www.nrl.navy.mil.itd.chacs/files/pdfs/13-1231-3743.pdf