Enhanced Performance and Privacy for TLS over TCP Fast Open

Erik Sy, Tobias Mueller, Christian Burkert, Hannes Federrath, and Mathias Fischer
University of Hamburg

Proceedings on Privacy Enhancing Technologies (PoPET) '20

2023.07.13.

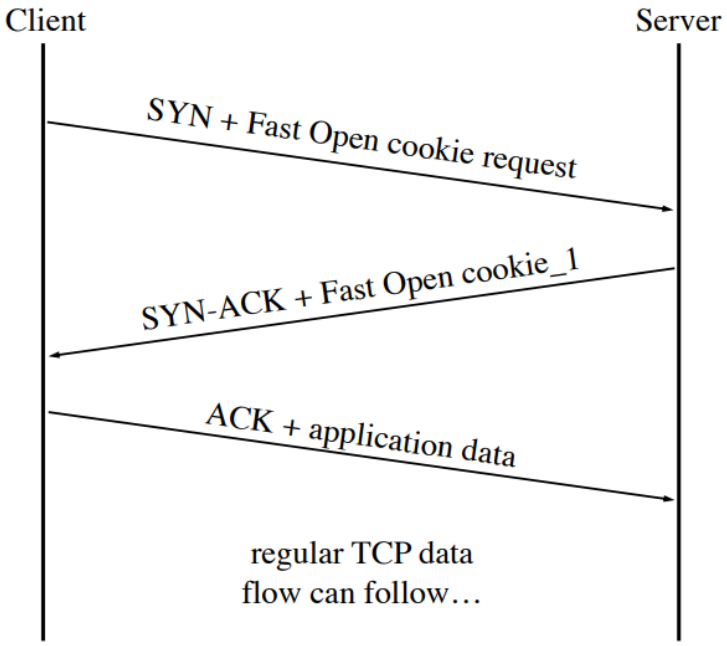
GyeongHeon Jeong(ghjeong@mmlab.snu.ac.kr)

Index

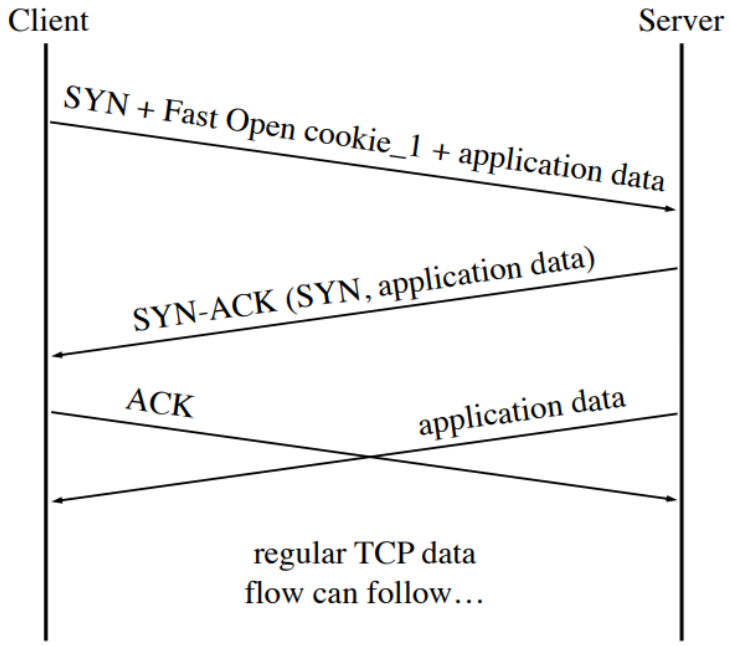
Introduction

- TCP is the standard network protocol for transmitting information on the Internet and is usually used to establish a subsequent TLS connection
- TCP Fast Open (TFO) protocol has been deployed to decrease the delay of a TCP handshake
  - Reduce TCP's three-way handshake to zero round-trip time (0-RTT) with **cookie**
  - Not yet actively used by all of operating systems and browsers
- Fast Open cookies can be used for **tracking** user
  - If IP addresses are not changed, cookies are perminent, and are also unencrypted
- Propose TCP Fast Open Privacy (**TCP FOP**) protocol as a countermeasure to TFO tracking

TCP Fast Open (TFO)



a) Initial Handshake



b) 0-RTT Handshake

```
Requesting Fast Open Cookie in connection 1:

TCP A (Client)                                          TCP B (Server)
______________                                          ______________
CLOSED                                                        LISTEN

#1 SYN-SENT       ----- <SYN,CookieOpt=NIL>  ---------->  SYN-RCVD

#2 ESTABLISHED    <---- <SYN,ACK,CookieOpt=C> ----------  SYN-RCVD
(caches cookie C)

Performing TCP Fast Open in connection 2:

TCP A (Client)                                          TCP B (Server)
______________                                          ______________
CLOSED                                                        LISTEN

#1 SYN-SENT       ----- <SYN=x,CookieOpt=C,DATA_A> ---->  SYN-RCVD

#2 ESTABLISHED    <---- <SYN=y,ACK=x+len(DATA_A)+1> ----  SYN-RCVD

#3 ESTABLISHED    <---- <ACK=x+len(DATA_A)+1,DATA_B>----  SYN-RCVD

#4 ESTABLISHED    ----- <ACK=y+1>--------------------> ESTABLISHED

#5 ESTABLISHED    --- <ACK=y+len(DATA_B)+1>----------> ESTABLISHED
```

Deployment of TFO

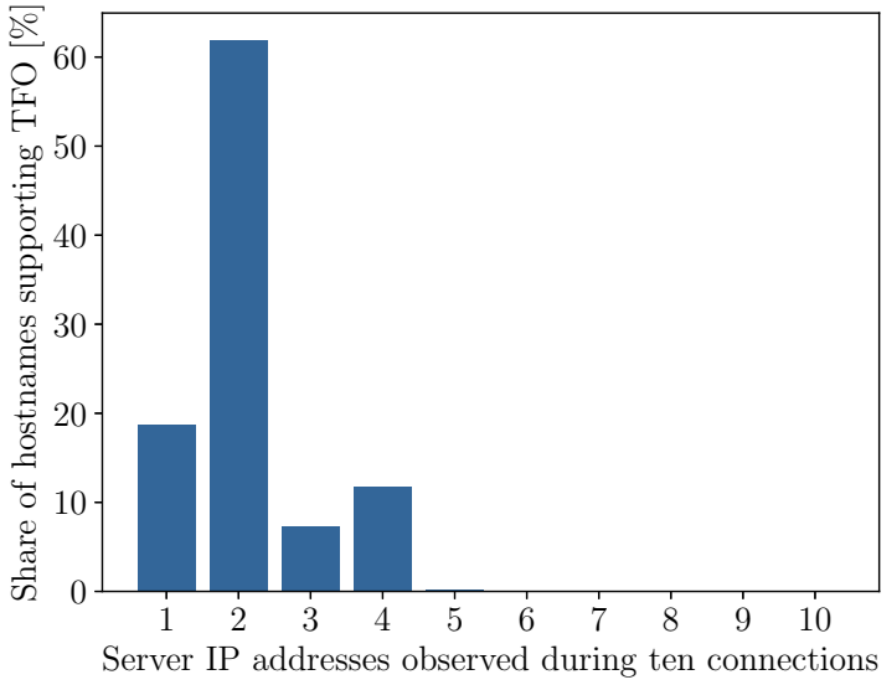
- Investigate the support for TFO within the Alexa Top Million Sites
- Higher-ranked websites tend to adopt new protocols such as TCP Fast Open earlier than other websites

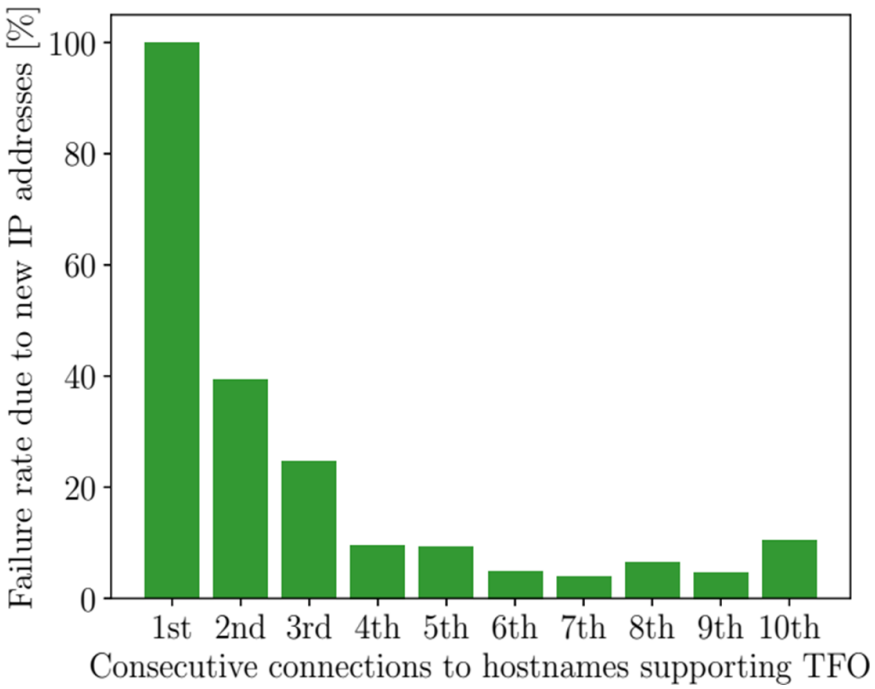
Table 1. Websites with TFO-support in Alexa Top lists

Alexa Top lists	Share of hostnames with TFO-support
Alexa Top 10	60.0%
Alexa Top 100	28.0%
Alexa Top 1K	12.4%
Alexa Top 10K	5.9%
Alexa Top 100K	3.4%
Alexa Top 1M	3.2%

Performance Limitations of TFO

- TFO protocol instructs to utilize a cached Fast Open cookie only if the source IP address, destination IP address, and the destination port match
  - But, due to server **load balancing**, connections to a hostname are not served from the same IP address everytime





Tracking via TFO

- Attacker model
  - Attacker is capable of extracting Fast Open cookies from the TCP headers, but cannot break cryptographic primitives

- **Third-party tracking**
  - The third-party can link website visits to the same user with their trackers

- **Tracking Across Virtual Domains**
  - Virtual hosting allows sharing resources like the IP address and server hardware
  - Operator of a virtual hosting platform can link visits of the same user across the hosted virtual domains

Evaluation of User Tracking via TFO

- Popular web browsers with various OS

Browser/Test system	Status	Tracking periods	Tracking across					
			Third-parties	Virtual hosts	IP addr. changes	Private browsing modes	User applications	Browser restarts
Chrome v68/Ubuntu 18.04	support	unrestricted	viable	viable	blocked	viable	viable	viable
Firefox v61/Ubuntu 18.04	support	unrestricted	viable	viable	blocked	viable	viable	viable
Firefox v61/macOS 10.13	support*	unrestricted	viable	viable	blocked	viable	viable	viable
Firefox v61/Windows 10	support*	unrestricted	viable	viable	blocked	viable	viable	viable
Edge v42/Windows 10	default	24 hours	viable	viable	blocked	viable	viable	viable
Opera v54/Ubuntu 18.04	support	unrestricted	viable	viable	blocked	viable	viable	viable

*Activated by default within Firefox Nightly and Firefox Beta.

Feasible Tracking Periods

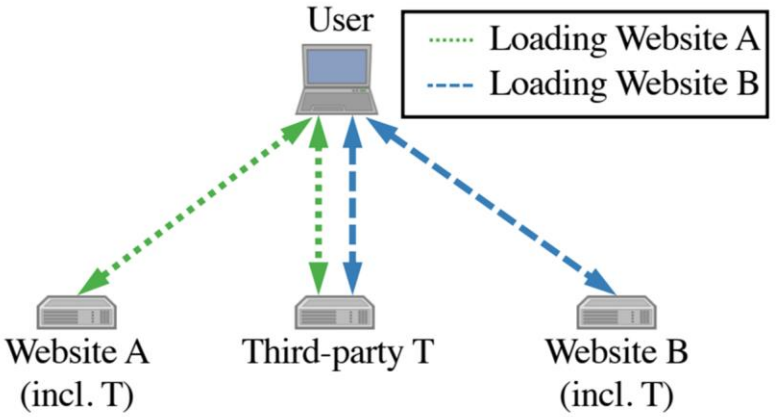
Browser/Test system	Status	Tracking periods	Tracking across					
			Third-parties	Virtual hosts	IP addr. changes	Private browsing modes	User applications	Browser restarts
Chrome v68/Ubuntu 18.04	support	unrestricted	viable	viable	blocked	viable	viable	viable
Firefox v61/Ubuntu 18.04	support	unrestricted	viable	viable	blocked	viable	viable	viable
Firefox v61/macOS 10.13	support*	unrestricted	viable	viable	blocked	viable	viable	viable
Firefox v61/Windows 10	support*	unrestricted	viable	viable	blocked	viable	viable	viable
Edge v42/Windows 10	default	24 hours	viable	viable	blocked	viable	viable	viable
Opera v54/Ubuntu 18.04	support	unrestricted	viable	viable	blocked	viable	viable	viable

- Visit a website that supports TFO and closed the browser tab, and after one hour (~ 10 days), revisit the same website with the same IP address
- For the Microsoft Edge browser, they were required to conduct this experiment on an IPv6 network stack
  - Use **temporary** IPv6 addresses which is limited to 24 hours by Windows 10

Tracking across Third-Parties

Browser/Test system	Status	Tracking periods	Tracking across: Third-parties	Virtual hosts	IP addr. changes	Private browsing modes	User applications	Browser restarts
Chrome v68/Ubuntu 18.04	support	unrestricted	viable	viable	blocked	viable	viable	viable
Firefox v61/Ubuntu 18.04	support	unrestricted	viable	viable	blocked	viable	viable	viable
Firefox v61/macOS 10.13	support*	unrestricted	viable	viable	blocked	viable	viable	viable
Firefox v61/Windows 10	support*	unrestricted	viable	viable	blocked	viable	viable	viable
Edge v42/Windows 10	default	24 hours	viable	viable	blocked	viable	viable	viable
Opera v54/Ubuntu 18.04	support	unrestricted	viable	viable	blocked	viable	viable	viable

- Visited website A and get cookie from the third-party T, then visited website B and check network traffic between the browser and T
- **None** of the tested browsers applied mechanisms to prevent third-party tracking



Tracking across Virtual Hosts

Browser/Test system	Status	Tracking periods	Tracking across					
			Third-parties	Virtual hosts	IP addr. changes	Private browsing modes	User applications	Browser restarts
Chrome v68/Ubuntu 18.04	support	unrestricted	viable	viable	blocked	viable	viable	viable
Firefox v61/Ubuntu 18.04	support	unrestricted	viable	viable	blocked	viable	viable	viable
Firefox v61/macOS 10.13	support*	unrestricted	viable	viable	blocked	viable	viable	viable
Firefox v61/Windows 10	support*	unrestricted	viable	viable	blocked	viable	viable	viable
Edge v42/Windows 10	default	24 hours	viable	viable	blocked	viable	viable	viable
Opera v54/Ubuntu 18.04	support	unrestricted	viable	viable	blocked	viable	viable	viable

- Visited web sites and then visit another web site which has the same IP address
- All investigated browsers **do not prevent** tracking across virtual hosts

Tracking across IP address changes

Browser/Test system	Status	Tracking periods	Tracking across					
			Third-parties	Virtual hosts	IP addr. changes	Private browsing modes	User applications	Browser restarts
Chrome v68/Ubuntu 18.04	support	unrestricted	viable	viable	blocked	viable	viable	viable
Firefox v61/Ubuntu 18.04	support	unrestricted	viable	viable	blocked	viable	viable	viable
Firefox v61/macOS 10.13	support*	unrestricted	viable	viable	blocked	viable	viable	viable
Firefox v61/Windows 10	support*	unrestricted	viable	viable	blocked	viable	viable	viable
Edge v42/Windows 10	default	24 hours	viable	viable	blocked	viable	viable	viable
Opera v54/Ubuntu 18.04	support	unrestricted	viable	viable	blocked	viable	viable	viable

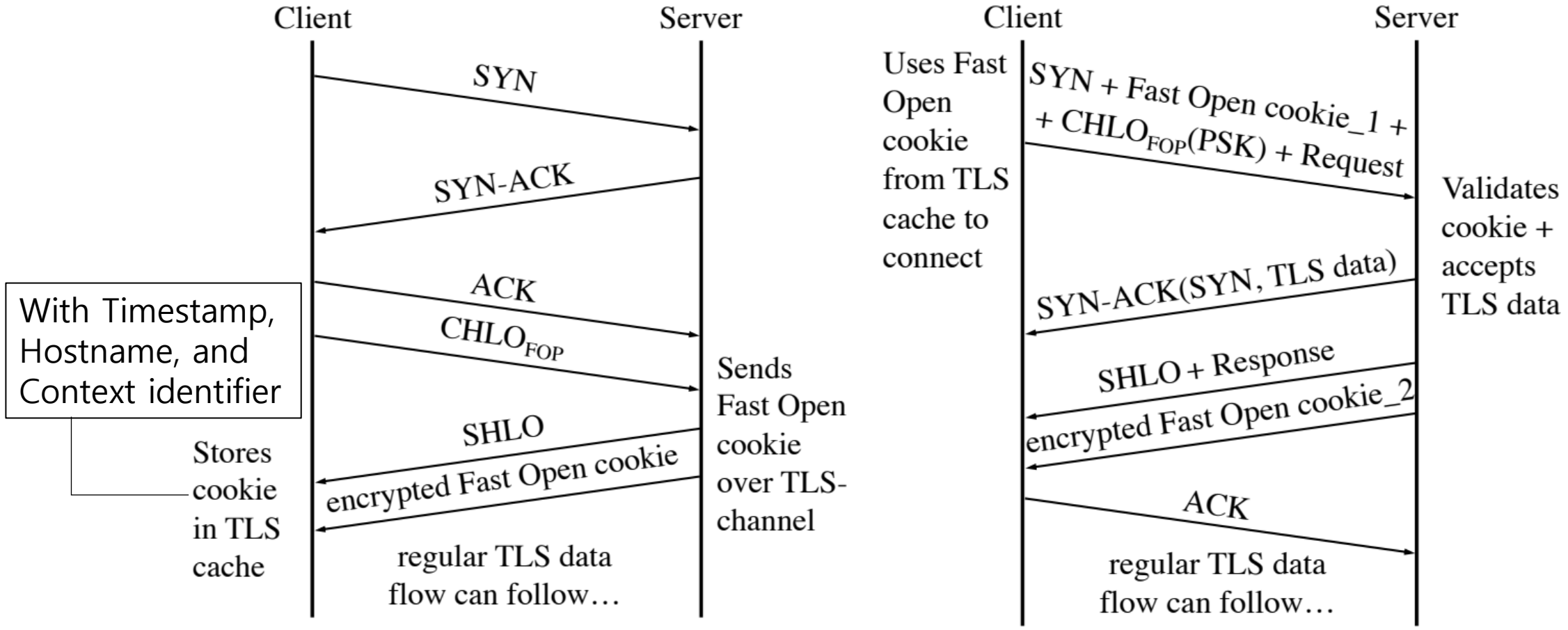
- Visited web sites and then visit another web site with another IP address
- However, considering a common consumer setup, devices typically keep their local IP addresses unchanged indefinitely
  - Since DHCP servers usually reassign the same local IP address based on client's MAC address

Tracking across Others

Browser/Test system	Status	Tracking periods	Tracking across					
			Third-parties	Virtual hosts	IP addr. changes	Private browsing modes	User applications	Browser restarts
Chrome v68/Ubuntu 18.04	support	unrestricted	viable	viable	blocked	viable	viable	viable
Firefox v61/Ubuntu 18.04	support	unrestricted	viable	viable	blocked	viable	viable	viable
Firefox v61/macOS 10.13	support*	unrestricted	viable	viable	blocked	viable	viable	viable
Firefox v61/Windows 10	support*	unrestricted	viable	viable	blocked	viable	viable	viable
Edge v42/Windows 10	default	24 hours	viable	viable	blocked	viable	viable	viable
Opera v54/Ubuntu 18.04	support	unrestricted	viable	viable	blocked	viable	viable	viable

- Re-connection with browser's private mode, restarted browser, other browser
- TFO protocol leads to huge privacy risks

TCP FOP (Fast Open Privacy)



a) Initial Handshake

b) 0-RTT Handshake using TLS Session Re-sumption

Privacy Evaluation of TCP FOP

- Server transmitted cookie with encrypted channel, and make new one in every connection
- Cookie stores with **timpstamp**, so cookie can be only used during certain time
- Cookie stores with **context identifier**, which defined the visited party, virtual host, IP address, browsing mode, user application, and browser session

Privacy characteristic	TCP Fast Open Protocol	TCP Fast Open Privacy Protocol
Tracking via network-based attacker	viable	blocked through single-use cookies & encrypted channel
Tracking across third-parties	viable	blocking possible through context identifier
Tracking across virtual hosts	viable	blocking possible through context identifier
Tracking across private browsing modes	viable	blocking possible through context identifier
Tracking across browser restarts	viable	blocking possible through context identifier
Tracking across user applications	viable	blocking possible through context identifier
Tracking across IP address changes	blocked	blocking possible through context identifier
Tracking periods	unrestricted	restriction possible through expiration of cookies

ALL BLOCKED or RESTRICTED

Performance Evaluation of TCP FOP

- Experiment using the TCP FOP Prototype
  - **Mean duration** to establish a connection between the client-server pair and to download a small website
  - TFO and TCP FOP have a computational overhead by generating, validating, and handling the cookies
  - Differences are **less than a millisecond** between them

Mean time (standard deviation)

Network	TCP/TLS		TFO/TLS		TCP FOP/TLS	
latency [ms]	Initial [ms]	Resumed [ms]	Initial [ms]	Resumed [ms]	Initial [ms]	Resumed [ms]
≈0.3	28.9 (3.6)	20.2 (2.7)	29.9 (3.5)	22.3 (2.9)	29.6 (3.6)	22.2 (2.9)
50 ms	189.8 (2.5)	132.6 (1.7)	190.0 (2.4)	83.7 (1.9)	190.0 (2.6)	83.8 (2.2)
100 ms	340.2 (2.1)	233.1 (1.4)	340.3 (2.1)	135.1 (1.6)	340.7 (2.1)	135.4 (1.6)
150 ms	490.3 (1.8)	332.9 (1.3)	490.7 (1.8)	185.3 (1.4)	491.1 (1.8)	185.7 (1.4)

Performance Evaluation of TCP FOP (cont.)

- Simulation considering Load-balancing
  - TFO/TLS attempt 0-RTT handshakes only when the server's IP address matches
  - TCP FOP/TLS attempt 0-RTT handshakes with matching hostnames
  - If real-world load-balancing of websites is considered, TCP FOP/TLS protocol stack significantly outperforms TFO/TLS

RTT of 60ms for the LTE connection

Simulation	1^{st} Revisit		2^{nd} Revisit		3^{th} Revisit	
	TFO/TLS	TCP FOP/TLS	TFO/TLS	TCP FOP/TLS	TFO/TLS	TCP FOP/TLS
Probability to save zero RTT	39.3%	0.0%	24.6%	0.0%	8.1%	0.0%
Probability to save one RTT	60.7%	0.0%	75.1%	0.0%	78.5%	0.0%
Probability to save two RTT	0.0%	100.0%	0.3%	100.0%	13.4%	100.0%
Mean delay overhead over LTE	-36.4 ms	-120.0 ms	-45.5 ms	-120.0 ms	-63.1 ms	-120.0 ms

Feasibility Analysis

- Middleboxes such as firewalls modify or block unfamiliar TCP packets
  - TFO has problem with it, but TCP FOP use standard TCP handshake in 1st connection
  - Because 0-RTT connection parts are same, so if middleboxes support TFO, they also support TCP FOP

- TCP FOP requires TLS libraries to control cookies
  - *"Kernel TLS"* already exists an implemented example for a performance-optimization causing a similar drawback between TLS libraries and Kernel functions

- Overall, TCP FOP provides sufficient performance and privacy benefits to justify a cross-layer solution

Kernel development community. Kernel TLS, 2019. URL www.kernel.org/doc/html/latest/networking/tls.html.

Conclusion

- This paper is the first to describe tracking via TFO cookies
- Under real-world conditions, the first revisit of a website supporting the TFO protocol fails in 40% of all cases
- Investigate the TFO configuration of popular browsers and found that the tracking periods for Chrome, Firefox, and Opera seem to be not restricted at all
- Propose TCP FOP to overcome the described privacy limitations of TLS over TFO
  - TCP FOP allows 0-RTT handshake for website revisits independently of the server's IP address

Thank you for listening