TLS → Post-Quantum TLS:

Inspecting the TLS landscape for PQC adoption on Android

Dimitri Mankowski,[1] Thom Wiggers,[2] Veelasha Moonsamy[1]

[1] Ruhr University Bochum, Germany
[2] PQShield, Netherlands

Gefördert durch
DFG Deutsche Forschungsgemeinschaft

RUHR UNIVERSITÄT BOCHUM RUB

CASA Cyber Security in the Age of Large-Scale Adversaries

HORST GÖRTZ INSTITUT

OUTLINE

Part I:

- Motivation
- Experiment
- Measurement Results

Part II:

- Impact on PQC
- Recommendations for:
  - Protocol designers
  - Developers
  - Android ecosystem

Part I

Motivation, Experiment, Measurement Results

PQShield

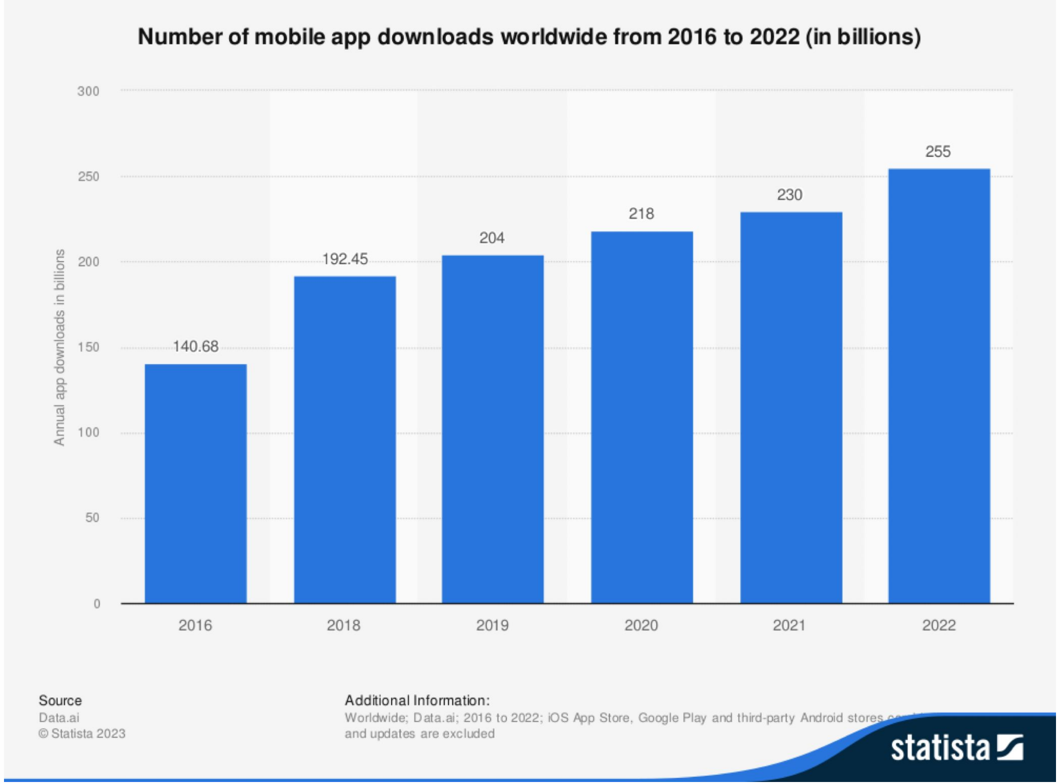
CASA
Cyber Security in the Age of Large-Scale Adversaries

Ruhr Universität Bochum RUB

Gefördert durch
DFG Deutsche Forschungsgemeinschaft

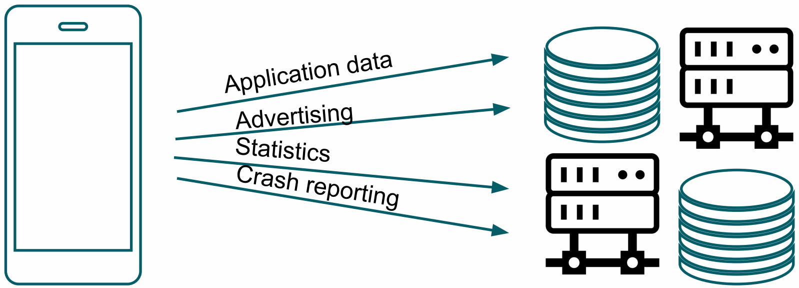
Horst Görtz Institut

APP USAGE



Number of mobile app downloads worldwide from 2016 to 2022 (in billions)
Annual app downloads in billions
300
250
200
150
100
50
0
140.68
192.45
204
218
230
255
2016
2018
2019
2020
2021
2022
Source
Data.ai
© Statista 2023
Additional Information:
Worldwide; Data.ai; 2016 to 2022; iOS App Store, Google Play and third-party Android stores
and updates are excluded
statista

MOTIVATION



Traffic encrypted with TLS

ECC and RSA → Efficient
Could be broken with quantum algorithms

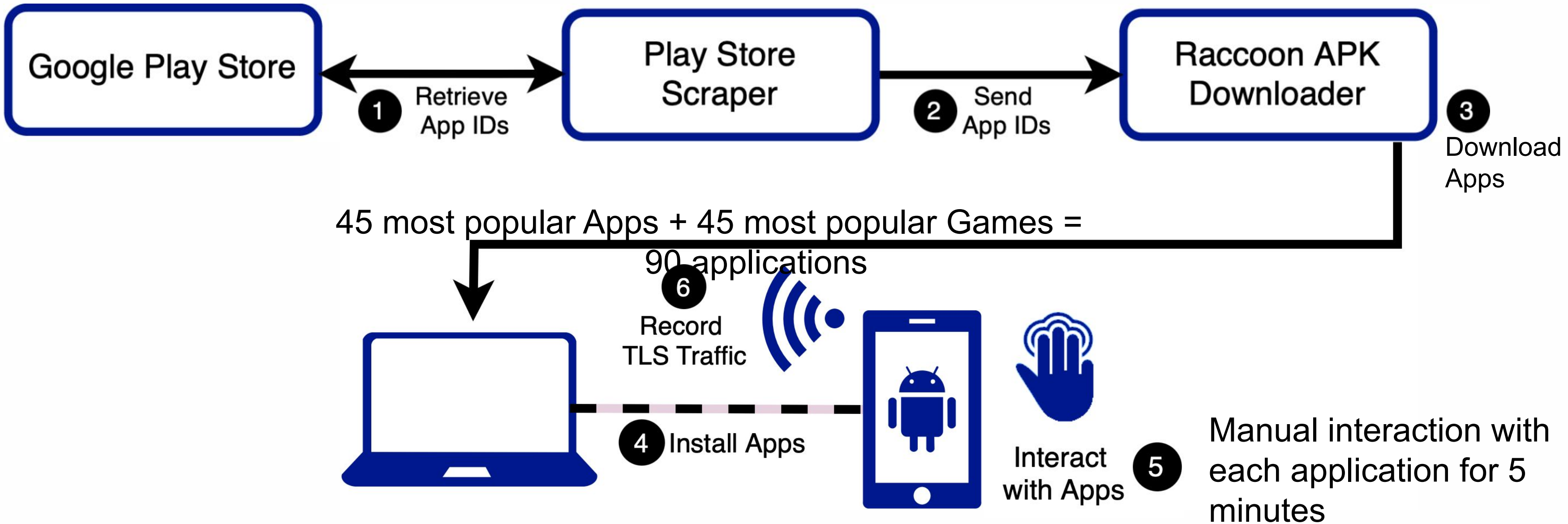
→

Use post-quantum cryptography (PQC)
Larger bandwidth requirements

How efficient would the adoption of PQC be in mobile apps?

Focus on Android → 78% market share worldwide
https://www.counterpointresearch.com/global-smartphone-os-market-share/

EXPERIMENT SETUP



45 most popular Apps + 45 most popular Games = 90 applications

Manual interaction with each application for 5 minutes

HOW TO REDUCE TLS OVERHEAD?

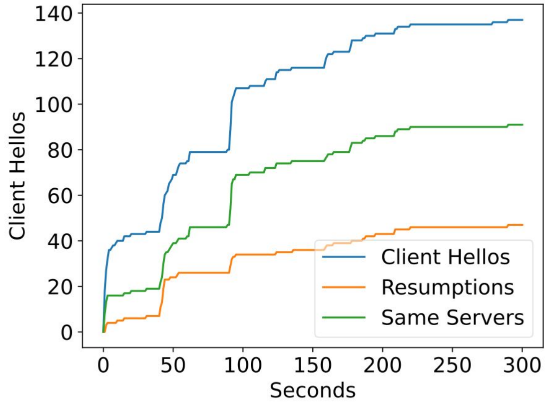
1. **Reduce number of handshakes**
   - Simplest way to reduce RTT
2. **Use Resumptions**
   - For repeatedly accessed servers
   - Re-establish a connection without performing a full TLS handshake
3. **Longer session durations**
   - e.g. HTTP Keep-Alive, HTTP/2 or HTTP/3 connection multiplexing
   - Could reduce the number of TLS handshakes
4. **TLS 1.3**
   - Reduces the number of round trips in handshake
   - Zero-round trip (0-RTT) mode for resumptions
5. **QUIC**
   - Uses TLS 1.3 and UDP, combining the connection setup and encryption handshake into a single round-trip

RESULTS - APPS VS GAMES

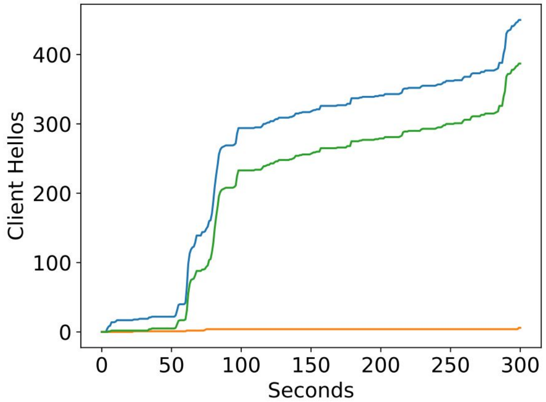
		Apps	Games	All
Handshakes	Mean	86	203	144
	Median	57	135	94
Resumptions	Mean	18	54	36
	Median	11	20	14
Servers	Mean	32	58	45
	Median	25	60	37
Traffic in MB	Mean	9.5	17.7	13.6
	Median	3.2	9	6.2
Session Time in secs	Mean	4.5	3.7	4.1
	Median	1.1	2.3	1.8
TLS 1.3 usage in %	Mean	73	58	66
	Median	77	67	69
QUIC handshakes	Mean	10	11	11
	Median	10	8	9

- **Games** are more active than **Apps** in almost all aspects
- Many **Apps** generate revenue through shopping/banking/…)
- **Games** generate revenue through advertising and data collection

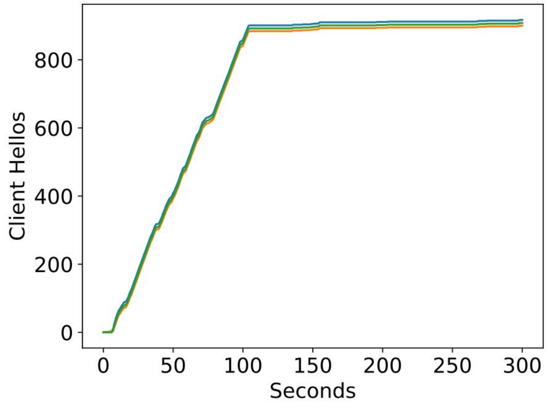
RESULTS - CLIENT HELLOS



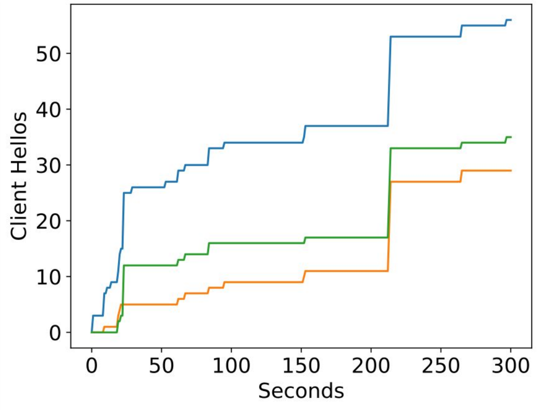
Amazon (App)



Roblox (Game)



Candy Crush Saga (Game)



Disney+ (App)

SUMMARY OF MEASUREMENTS

- **Slow** adoption of **new TLS standards** among Android applications
- **TLS 1.3** only used in **66%** of connections
- Only **31%** of connections to the same host use **resumptions**
- Use of the **QUIC** protocol remains **low**
- **Conclusion**: Focus of developers is largely not on network optimization

Part II

The impact of post-quantum cryptography

PQSHIELD

CASA
Cyber Security in the Age of Large-Scale Adversaries

RUHR UNIVERSITÄT BOCHUM RUB

Gefördert durch
DFG Deutsche Forschungsgemeinschaft

HORST GÖRTZ INSTITUT

POST-QUANTUM CRYPTOGRAPHY

POST-QUANTUM TLS

- Replace elliptic-curve Diffie–Hellman by post-quantum key exchange (KEM)
- Replace RSA/ECDSA by post-quantum signature schemes

NIST PQC standardization competition “winners”:

- KEM: **Kyber** (MLWE-KEM)
- Signatures: **Dilithium** (MLWE-Sign) (“primary” selected algorithm)

Algorithm	**public key size**	**ciphertext/signature**
Kyber-512 (KEM)	800 bytes	768 bytes
Dilithium-2 (Signature)	1312 bytes	2420 bytes

INCREASES IN SIZE

Ephemeral key exchange	**TLS handshake data** 1x public key + 1x ciphertext	**Authentication signatures**	**TLS handshake data** 2x public key + 3x signature
ECDH (X25519)	64 bytes	RSA-2048	1312 bytes
Kyber-512	**1568 bytes**	Dilithium2	**9984 bytes**

EXTRAPOLATING APP TRAFFIC

App	# Full HS	Key exchange	Data	Auth.	Data	Total crypto overhead
Klarna	51	X25519 Kyber-512	3.3 80.0	RSA-2048 Dilithium2	66.9 504.1	70.2 584.1
Lighter Simulation	257	X25519 Kyber-512	16.4 403.0	RSA-2048 Dilithium2	337.2 2540.2	353.6 2943.2
Haircut prank, air horn & fart	320	X25519 Kyber-512	20.5 501.8	RSA-2048 Dilithium2	419.8 3162.9	440.3 3664.6

REDUCING TLS IMPACT

Alternative proposals for more efficient post-quantum TLS:

- **KEMTLS**: use (smaller) post-quantum KEM instead of signatures for handshake authentication
- **KEMTLS-PDK**: supply TLS client with server KEM public key (e.g. by embedding in statistics/ads SDK) and use that to avoid server certificates entirely.

[KEMTLS]: Peter Schwabe, Douglas Stebila, Thom Wiggers (2020). Post-Quantum TLS without handshake signatures. ACM CCS 2020.
[KEMTLS-PDK]: Peter Schwabe, Douglas Stebila, Thom Wiggers (2021). More efficient post-quantum KEMTLS with pre-distributed public keys. ESORICS 2021.

ALTERNATIVE TLS HANDSHAKES

Handshake	Algorithms	Size of public key crypto (bytes) KEX	Auth.	Sum
TLS	Kyber-512 & Dilithium2	1568	9884	11 452
KEMTLS	Kyber-512 & Dilithium2	1568	7720	9288
KEMTLS-PDK	Kyber-512	1568	768	2336
KEMTLS-PDK	Kyber-512 & McEliece348864	1568	96	1664

CONCLUSIONS AND RECOMMENDATIONS

- Android apps set up **a lot of TLS connections**
- Techniques that reduce overhead of TLS are **hardly used**
- Transitioning to post-quantum security will **greatly increase impact of overhead**
- Pursuing alternatives to the signed-TLS handshake, especially KEMTLS-PDK, may be worthwhile

Recommendations

- **For protocol designers:** advanced features work, but developer visibility is an issue
- **For developers:** Adopting QUIC / TLS resumption / HTTP/2 / HTTP/3 today will greatly ease transition to post-quantum security tomorrow
- **For the Android ecosystem:**
  - Improve documentation and default library settings to encourage using the above
  - Give developers tools to inspect their apps' TLS usage (as browsers do!)

Paper available at:
https://ia.cr/2023/734
Dataset and scraper available at:
https://zenodo.org/record/7950522

Thanks for your attention