WEBINAR SUMMARY

March 25, 2021

The State of State Machines

Natalie Silvanovich, Security Researcher, Google Project Zero
Hank Schless, Senior Manager, Security Solutions, Lookout

KEY TAKEAWAYS

- WebRTC: Understanding the most common audio and video conferencing library.
- Video conferencing state machine security bugs are common, but preventable.
- Phishing attacks can come from anywhere, including mobile applications.
- Mobile application vulnerabilities drove a spike in phishing attacks in 2020.
- Organizations need to face the challenges posed by mobile phishing.

in partnership with

Lookout

OVERVIEW

In January 2019, a serious vulnerability was found in Apple's FaceTime application, allowing an attacker to call a target device and force the call to connect without user interaction. This logic bug in the application's calling state machine allowed the attacker to listen in to the target's surroundings without their knowledge or consent.

State machine vulnerabilities remain a problem in other mobile applications, allowing attackers to gain access to audio and video. Mobile phishing attacks also take advantage of issues within the state machine, allowing attackers to gain anything from credentials to remote control of the device.

CONTEXT

Natalie Silvanovich shared her research into state machine vulnerabilities with video conferencing applications using a common technology, Web Real-Time Communication (WebRTC). Hank Schless discussed how mobile application vulnerabilities are driving increased phishing attacks on mobile devices.

KEY TAKEAWAYS

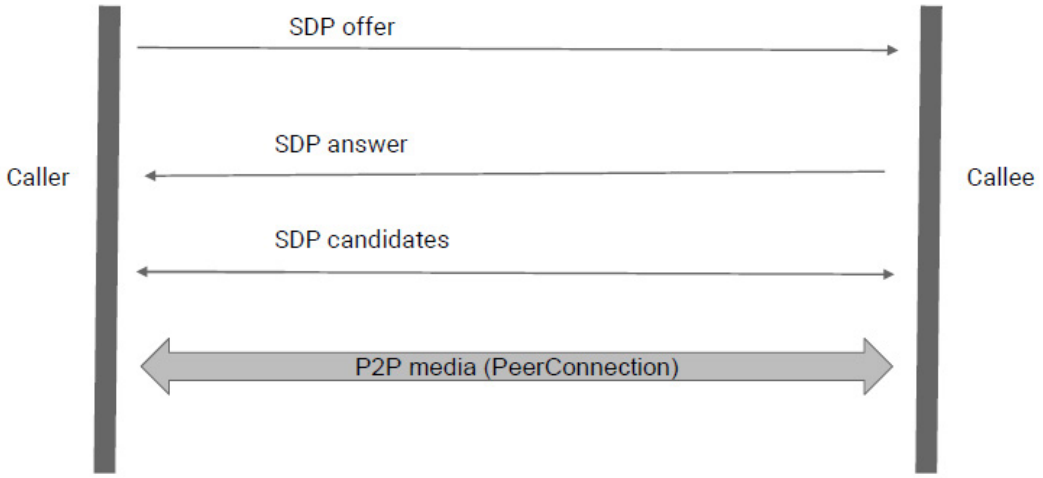
WebRTC: Understanding the most common audio and video conferencing library.

While the FaceTime bug may be the best known of its kind, similar security bugs are common in other video conferencing applications, many of which use WebRTC. This communications technology is an audio and video conferencing library maintained by Google that is used by all major web browsers and many mobile Android and Apple iOS applications, including Signal Messenger and Facebook Messenger.

Understanding how signaling states work in WebRTC and the design decisions that need to be made in implementation can help developers avoid security problems. As Figure 1 shows:

1. The caller system will send a session description protocol (SDP) offer, which contains information necessary to set up the call, including internet protocol address on the network, encryption keys, and codec keys.
2. The callee sends an SDP answer in the same format as the offer, which can include any changes to the technologies used.
3. The caller and callee, based on what is agreed upon in the offer and response, gather SDP candidates.
4. The SDP candidates are then shared, so that the peer-to-peer connection can be made.

Figure 1: The call signaling flow when a video conferencing call is made



What is being sent through the stream—audio, video, or both—is defined using tracks that are added to a PeerConnection and enabled before the input is streamed. Tracks are input devices that can be streamed to a peer: camera and microphone. This can be done at any time during a call, but transmission won't work until a PeerConnection has been established.

Video conferencing state machine security bugs are common, but preventable.

Video conferencing signaling state bugs are common, and have been found and fixed in some well-known applications. Most of these problems are preventable, occurring in how the developers design calling state machines and not within WebRTC design or documentation.

Table 1: Examples of found and fixed vulnerabilities in video applications using WebRTC

Application	Vulnerability	Root Cause
Signal Messenger	Transmitted audio without consent.	Confusion between the caller and callee state.
Facebook Messenger	Transmitted audio without consent.	State mismanagement; didn't check the state between the caller and callee, and also didn't check whether the call was in progress, so the message type was passed before the call, rather than after it was established.
JioChat and Mocha	Transmitted audio and video without consent.	Misunderstood that offers and answers can contain candidates.
Google Duo	Transmitted a few frames of video without consent.	The state machine worked, but asynchronous logic caused a race condition.

When designing the implementation, there are three design choices that need to be made. It is more secure to do all of these after the call is accepted, but this can cause performance issues, such as a delay in the connection.

Design choices for implementing WebRTC

- When to exchange the offer and answer:
  - Both are exchanged before the user picks up the call.
  - Both are exchanged after the user picks up the call.
  - Only the offer is exchanged before the user picks up, and the answer is exchanged after the user picks up.
- When to exchange SDP candidates.
- When to create, add, and enable tracks.

> It's more secure to do everything after the callee has accepted the call. The more that's done before, the more attack surface is available.
>
> *Natalie Silvanovich, Google Project Zero*

Phishing attacks can come from anywhere, including mobile applications.

Phishing attacks don't just come from emails; on mobile devices they can, and usually do, come from other applications. In 2020, 85% of mobile phishing attacks happened on applications outside of email.

These phishing attacks are often harder to combat on mobile devices than they are on a computer:

- **Mobile user interface design hides details** that are typically visible and help identify a phishing attack on a computer; for example, hovering over a web address to see whether it is known or trustworthy.
- **Social engineering on mobile devices is easier,** especially through the messaging components available in many social media apps, dating apps, and gaming apps.

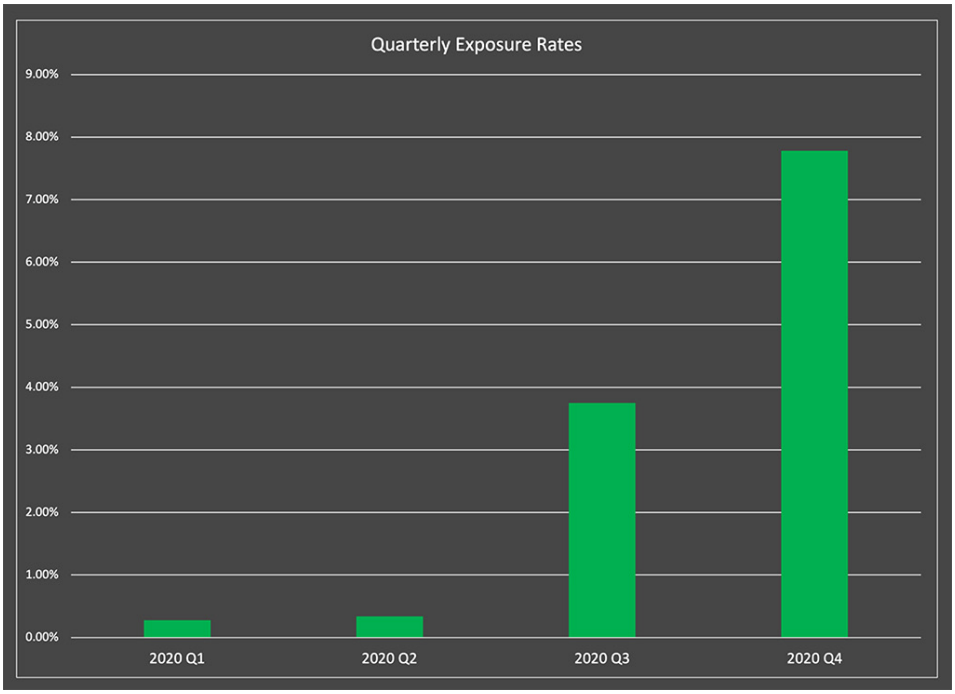
> We see the mobile device as an extension of ourselves and we trust ourselves to be safe, so it's natural to think devices are inherently secure even when it's been proven otherwise.
>
> *Hank Schless, Lookout*

Although it is more difficult to do, not protecting against mobile phishing leaves a significant gap in an organization's security posture. In 2020, 11% of enterprise devices encountered a phishing link, 43% of which were built to harvest credentials and 80% of which were built to deliver malware.

Mobile application vulnerabilities drove a spike in phishing attacks in 2020.

Vulnerabilities in major mobile applications, including Google Chrome, Instagram, and the SourMint software developers kit, drove a significant increase in mobile phishing attacks in the second half of 2020.

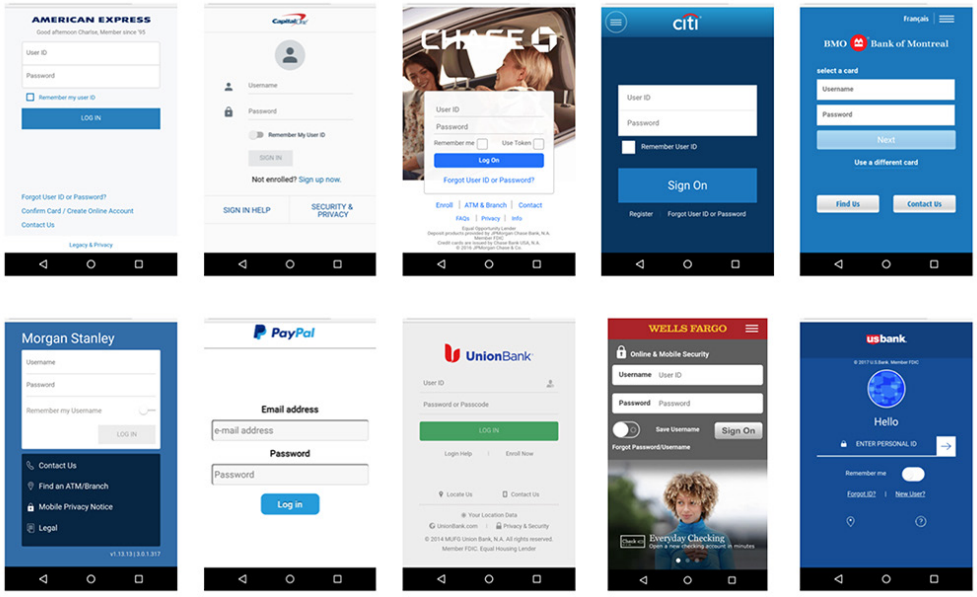
Figure 2: Exposure to malicious and risky mobile apps spiked in the second half of 2020



In additional to individual exploits, malware-as-a-service, which is available to numerous attackers, continues to be a problem for mobile devices. The number of observed samples of BancaMarStealer, for example, has grown from 7,700 in 2018 to more than 74,000 in 2020.

BancaMarStealer, which is delivered by short message service (SMS), prompts the target to download a malicious app, often made to look like a trusted banking application, although samples have been analyzed that overlay Amazon, Facebook, Skype, Twitter, Uber, and WhatsApp. The malware can harvest credentials, send the user to other malicious sites, retrieve all SMS, and take control of the device remotely.

Figure 3: Malware-as-a-service BancaMarStealer uses banking application overlays in its phishing attacks



Organizations need to face the challenges posed by mobile phishing.

Mobile phishing is hard to combat, especially with numerous mobile devices used across an organization. Businesses need to be aware of these devices and the challenges they face with mobile phishing.

Challenges Created by Mobile Phishing

- It is difficult to have visibility into every application, permission, and risk across the organization, especially in employee-provided devices.
- Without the right setup, organizations will need to rely on individuals to update their devices.
- There is often a lag time in developers issuing patches, and an even longer lag time in when users update their devices.
- If it is an on-premise/on-device vulnerability, patches can't be pushed through cloud updates.
- It is difficult to implement zero trust security without knowing each device and its risks.

ADDITIONAL INFORMATION

- For more information on the state of state machines, visit the Google Project Zero team's blog.
- For more information on Lookout and its findings:
  - Visit Lookout's application threat map and phishing threat map
  - Read When legit apps turn malicious. (It happens more often than you think.)
  - Read Lookout Discovers Novel Confucius APT Android Spyware Linked to India-Pakistan Conflict
  - Learn about Lookout's acquisition of CipherCloud for cloud-delivered security

BIOGRAPHIES

Natalie Silvanovich

Security Researcher, Google Project Zero

Natalie Silvanovich is a security researcher on Google Project Zero. Her current focus is on script engines, particularly understanding the subtleties of the scripting languages they implement and how they lead to vulnerabilities. She is a prolific finder of vulnerabilities in this area, reporting over a hundred vulnerabilities in Adobe Flash in the last year.

Previously, she worked in mobile security on the Android Security Team at Google and as a team lead of the Security Research Group at BlackBerry, where her work included finding security issues in mobile software and improving the security of mobile platforms. Outside of work, Natalie enjoys applying her hacking and reverse engineering skills to unusual targets and has spoken at several conferences on the subject of Tamagotchi hacking.

Hank Schless

Senior Manager, Security Solutions, Lookout

After working for fintech and database technology companies, Hank has found a home in cybersecurity. Before Lookout, he was the 20th employee at a cloud infrastructure security startup and helped the company grow to over 160 employees. At Lookout, he is a Senior Manager on the Security Solutions team, enabling internal teams and informing the market about the growing need to secure mobile devices as part of the larger security strategy.