I Got 99 Problems but an 0Day Ain't One: Inside the Campaign Exploiting iPhones and Android Devices in APAC

John Bambenek

ILLINOIS

About Me

- PhD Student in Informatics at the University of Illinois at Urbana-Champaign
- VP of Security Research and Intelligence at ThreatSTOP
- Produce and Sell Open-Source Threat Feeds as Bambenek Consulting, LTD.

Sharing Restrictions

- The content of this talk can be considered TLP:WHITE, share/tweet away.

Honorable Mentions

- This was found by Google Project Zero in the first place.
- Volexity has done great follow on work with attribution and on the Android pieces of this.
- Citizen Lab identified a similar campaign against Tibet.
- Several others looked at this as well and contributed.

Dishonorable Mention

- Apple’s Response:
  - “First, the sophisticated attack was narrowly focused, not a broad-based exploit of iPhones 'en masse' as described. The attack affected fewer than a dozen websites that focus on content related to the Uighur community…”
- There are lots of good ways to respond to vulnerabilities in your devices… this was not one of them.

Agenda

- Victimology / Attribution
- C2 infrastructure
- iPhone exploits
- Android exploitation
- Malware

Victimology / Attribution

- So far, these attacks originate from websites typically read by Uighur Muslims or Tibetans.
- Thought to be possibly 2 different APT groups using these attacks.
- CitizenLab has called one of them Poison Carp. Volexity calls them Evil Eye.
  - Scanbox was observed briefly, which may imply the Comment Crew involvement.
- Groups almost certainly sponsored or supported by the PRC.

Websites compromised

- Akademiye.org
- Turkistantimes.com
- Uighurtimes.com
- Wewuer.com
- Iuyghur.com
- Istiqlalhaber.com
- Turkistanpress.com
- Turkistantv.com
- Maarip.org

- Tibetan campaign used malicious links sent to office of Dalai Lama, Tibetan Parliament, and Tibetan Administration.

- Implants were related suggesting same group (or closely related groups).

Compromised Websites

- Typically used an invisible iFrame on otherwise valid sites.
- Android attacks used typo-squatted names.
- Some compromised pages also had “fake oauth” login pages to steal credentials.
- There was analysis of visitors to determine suitability with blacklisting.
- FBI reportedly asked Google not to index some of these pages.

C2 Infrastructure

- www.google-analysis.info
- turkistantlmes.com
- akademlye.org
- ajax.cloudflarestatic.tk
- **app.msap.services**
- arkinixik.ezua.com
- stats.uyghurmedia.top
- getip.name
- emailgroup.uyghurmedia.top
- d.scanvpn.com

- 182.61.106.160
- 182.61.189.138
- 149.28.207.244
- 144.202.59.23
- 149.248.57.231
- 139.180.223.184
- 150.109.120.186
- 45.32.190.160
- 142.4.50.213
- 182.61.184.33
- 182.61.171.167
- 182.61.173.209
- 182.61.176.128
- 45.78.79.100
- 149.28.93.11
- 95.169.2.57
- 206.189.65.198
- 140.82.17.222
- 45.32.91.137

Choopa – US
IT7 Networks – CA
Were most common

No previous indications of maliciousness

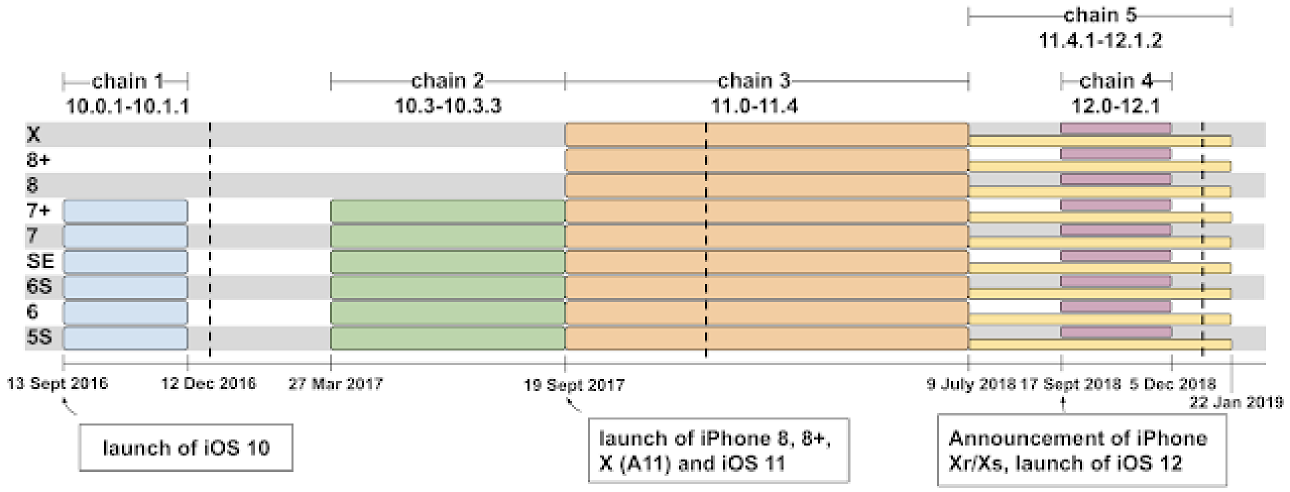
C2 Infrastructure

- Whois:
  - ornaments798@outlook.com
  - dashenqu832@outlook.com

- Oauth indicators:
  - antmoving.online@gmail.com
  - energymail.org@gmail.com
  - jameslewis199106@gmail.com
  - touchxun658@gmail.com
  - uygurye2008@gmail.com

iPhone exploits

- Originally reported by Project Zero.
- 5 different chains of 14 exploits (7 for Safari, 5 for kernel, 2 sandbox escapes) effected up to iOS 12.1.2
- Primarily using JavaScript within WebKit.



iPhone exploits

- Google reported the exploitation window from time of deployment of iOS version to time of patch, Apple disputes.
  - Vulnerability likely not available on 1st day of version release.
- What it does show is attacker kept adapting to updates.
- Users don’t upgrade their phones right away.

Exploit chain 1

- Vulnerable versions: 10.0.1 – 10.1.1

- Heap overflow in AGXAllocationList2::initWithSharedResourceList which is part of driver for GPU in iPhone.
  - Takes two arguments, one of which is a resource list header which is attacker controlled.
  - Parsing code makes assumption there are only 6 headers but there is space for 7.

Exploit chain 1

- They use it to allocate two shared memory regions and then start 100 threads.
  - Used for heap grooming
- They abuse how mach messages can contain “out-of-line data”.
  - Lets them overwrite pointers.
- They also iterate over the proc structure to overwrite the the ucreds label to “unsandbox” the process.

Exploit chain 1

- Relies on kernel task port, which gives memory read-write access to anyone with the rights for it.
- Attacker corrupted memory to give them this right which allows for modifying kernel memory.

Bypassing Policy

- Every iOS device has a platform policy in com.apple.security.sandbox.kext which is protected.
  - The pointer to that policy, however, is not.
- Attacker changes the pointer and removes the policy that prevents process execution.
- The kernel also has an array of implicitly trusted code signing blobs which is accessible to those with kernel-write.
  - They just add their signing blob to the array which allows their code to execute.
  - Recent jailbreaks use a similar approach.

Exploit chain 2

- Vulnerable versions: 10.3 – 10.3.3

- Another kernel bug (CVE-2017-13861), essentially a use-after-free vulnerability between IOSurfaceRootUserClient and IOConnectCallAsyncMethod. Would drop wake_port twice and make reference count wrong.

- New mitigation for kernel task ports, but easily bypassed by making a copy at a different kernel address.

Exploit chain 2

- Exploitation process now rewrites ucred to original values after process has spawned.
- They also create a flag (“iopl”) that the device is exploited that they check for to avoid re-exploitation.
- Boot arguments (where they store the flag) can be read inside the Safari sandbox.

Exploit chain 3

- Vulnerable versions: 11.0 – 11.4.1
- Uses a new boot argument “iopl14” for infection.
- Relies on a bound-checking error (changing a < to !=) in libxpc that helped the attacker escape the sandbox. Developers refactored code that in-effect broke the bounds checking by comparing to a single invalid value.
- The attackers heap spray and target mediaserverd (because it can access vulnerable IOKit driver).

Exploit chain 4

- Vulnerable versions: 12.0-12.1
- Uses cfprefsd daemon which is unsandboxed and performs an unbounded memmove with a length argument controllable by the attacker. This daemon incorrectly uses XPC which allows it to be exploited.
- They then heap and port spray to exploit the kernel.
- Changed compromise flag to check maxfilesperproc setting.

Exploit chain 5

- Vulnerable versions: 11.4.1-12.1.1
- Uses only one vulnerability and more reliable than chain 4.
- Apple partially created a feature *and deployed the partial code* for "vouchers. The idea was for a new syscall but it never worked and if you tried to use it, it would crash the phone.

- They create before, target, and after vouches and destroy them, but the threat still points to the "freed" target voucher.

- The convert voucher to a port, do another round of voucher create and destruction and ultimately create a fake kernel task in the buffer with read/write.

Android Exploit

- Used hidden iframes in compromised websites (sometimes with very similar domain names).
  - Have code that gets browser to open crafted website with exploit.

- Exploits span chrome 39 to 73 (with gaps) and 8 different exploits. First seen in 2014.
  - Exploitation reduced to one-click.

- Each exploit ran the same shellcode and downloaded the implant (even through other applications).

iPhone Malware

- Had access to all backend databases on the phone.
  - Included photos, GPS, contacts
  - Encrypted messaging apps store messages in clear in those databases.
  - Would upload sqlite databases only if on wifi
  - Would also get the keychain (with creds)
  - Also took “long-lived” oauth tokens so they could get access to services independently (i.e. gmail)
- Apps targeted:
  - Viber, Voxer, Telegraph, Gmail, Twitter, QQMail, WhatsApp, **Yahoo Mail, Outlook, NetEase Mail Master, Skype, Facebook, WeChat**
- Would beacon to C2 server over HTTP in clear to exfiltrate data.

Android Malware

- Dubbed MOONSHINE by Citizen Lab consisting of three stages.
- **Whisky** – Initial downloader, tries to unpack libbourbon.so to overwrite a legit shared library (and maintain persistence).
  - Looks for Facebook, Facebook Messenger, WeChat, or QQ.
- **Bourbon** – Started as part of the app, loads Scotch java app.
- **Scotch** – Does beaconing and exfil.

- Uses hard-coded user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:65.0) Gecko/20100101Firefox/65.0
- Uses a unique IP checking utility getip.name which is no longer online.

Oauth targeting

- As part of this ongoing campaign, emails, WhatsApp, and Facebook messages were sent pointing people to typosquated websites trying to steal get users to give them third party “app” access to read their email.

- One posed as a journalist to a member of Tibet’s parliament asking for confirmation on a story.

Closing

- After Volexity, P0, CitizenLab reporting, most of this was cleaned up and removed.
- Similar campaigns have exploited PCs for more traditional intelligence collection. The targeting of phones highlights the physical risk to the victims.
- Android campaigns as early as 2014, iPhone as early as 2016. This group is spending real effort developing tools and exploits. Periods where they had no “actionable” exploits but they kept working.

Closing

- Android malware used hard-coded user-agent and can be used for detection, but likely this will be changed now that it’s been reported on.

- Some sites used Let’s Encrypt for SSL/TLS certs. Using certificate transparency logs with “fuzzy” matching on keywords would provide more persistent proactive detection.

Questions

- bambenek@Illinois.edu / jbambenek@threatstop.com
- @bambenek