Chamois

Android's Most Impactful Botnet of 2018

Maddie Stone

@maddiestone

Kaspersky Security Analyst Summit 2019

Who am I? - Maddie Stone

- Senior Reverse Engineer and Tech Lead on Google Play Protect
- 6+ years hardware & firmware reversing
- Speaker at REcon, OffensiveCon, BlackHat, & more!
- BS in Computer Science, Russian, & Applied Math, MS in Computer Science

@maddiestone

Chamois

Sophisticated botnet that backdoors applications to do ad fraud, SMS fraud, and install fraud.

Android Application Ecosystem

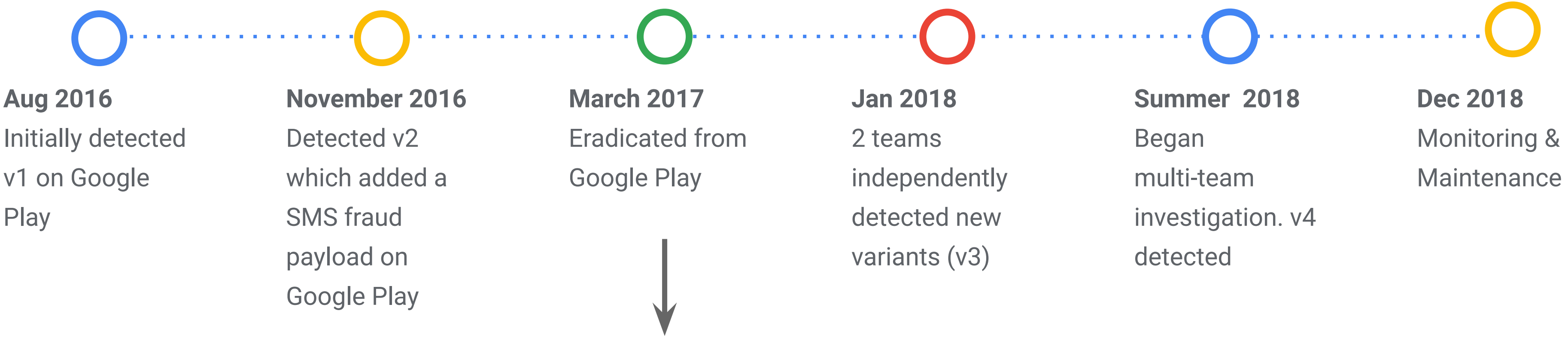
- Distribution of Apps
  - Google Play store vs not Google Play (sideloaded, 3P app stores, pre-installed)
- PHA = Potentially Harmful Applications
- GPP = Google Play Protect
- APK = Android App

Overview

- PHA Category: Backdoor
- Initially detected in Mid-2016
- SDK that third-party app developers package into their apps (unknowingly that it’s a botnet backdoor)

- 4 distinct variants
- All variants contain 4-6 stages
- Payloads:
  - Premium SMS fraud
  - App install fraud
  - Ad fraud
  - Arbitrary module loading

Timeline



Aug 2016
Initially detected v1 on Google Play

November 2016
Detected v2 which added a SMS fraud payload on Google Play

March 2017
Eradicated from Google Play

Jan 2018
2 teams independently detected new variants (v3)

Summer 2018
Began multi-team investigation. v4 detected

Dec 2018
Monitoring & Maintenance

Google Blogpost
Detecting and eliminating Chamois, a fraud botnet on Android

Most impactful?

- Technical complexity
- Multiple distribution channels
- Rapid and mature release process
- Actor has resources: technical expertise, funding, infrastructure, etc.
- Advanced ad fraud techniques

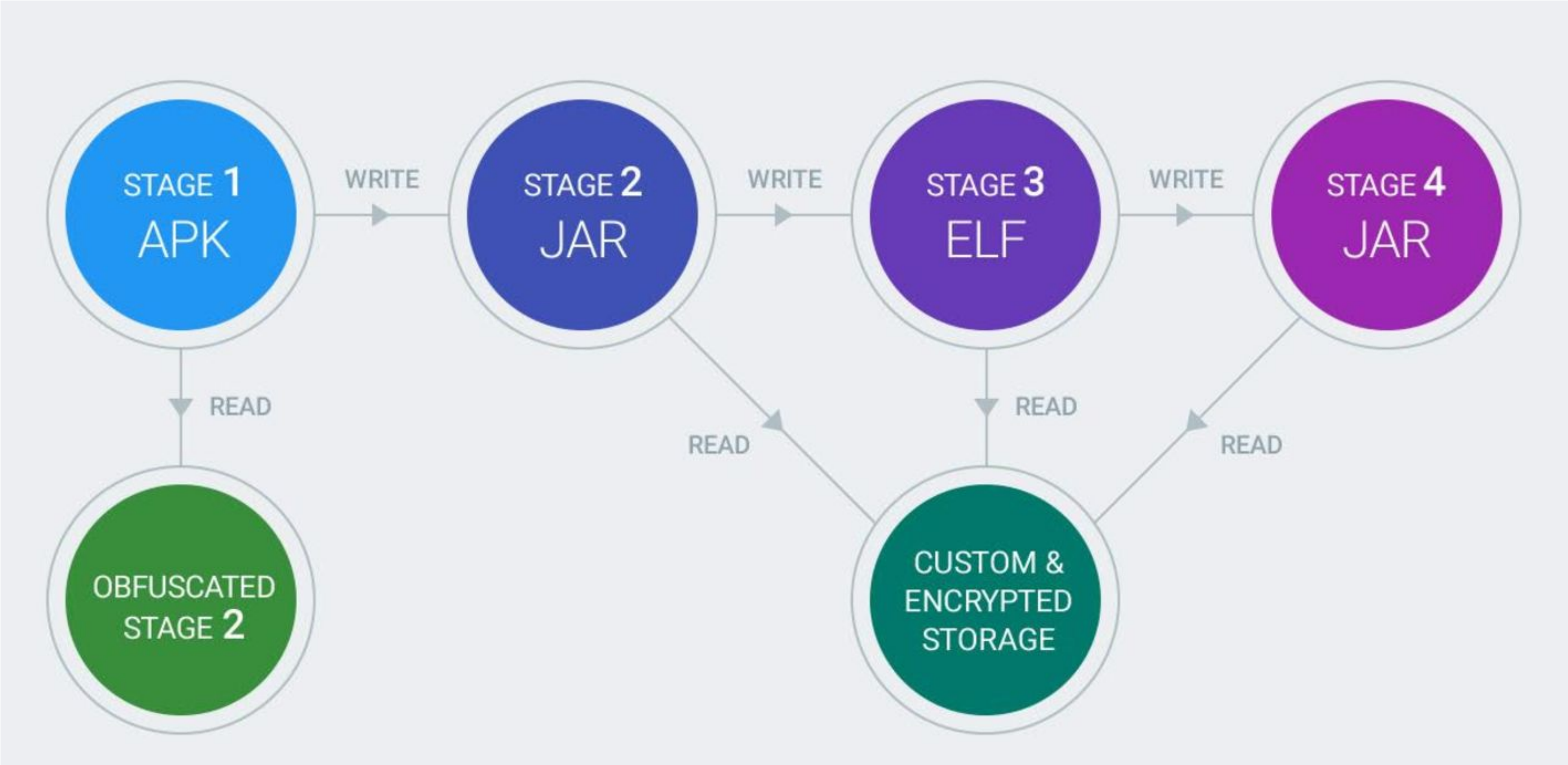
Technical Details

Variants Overview

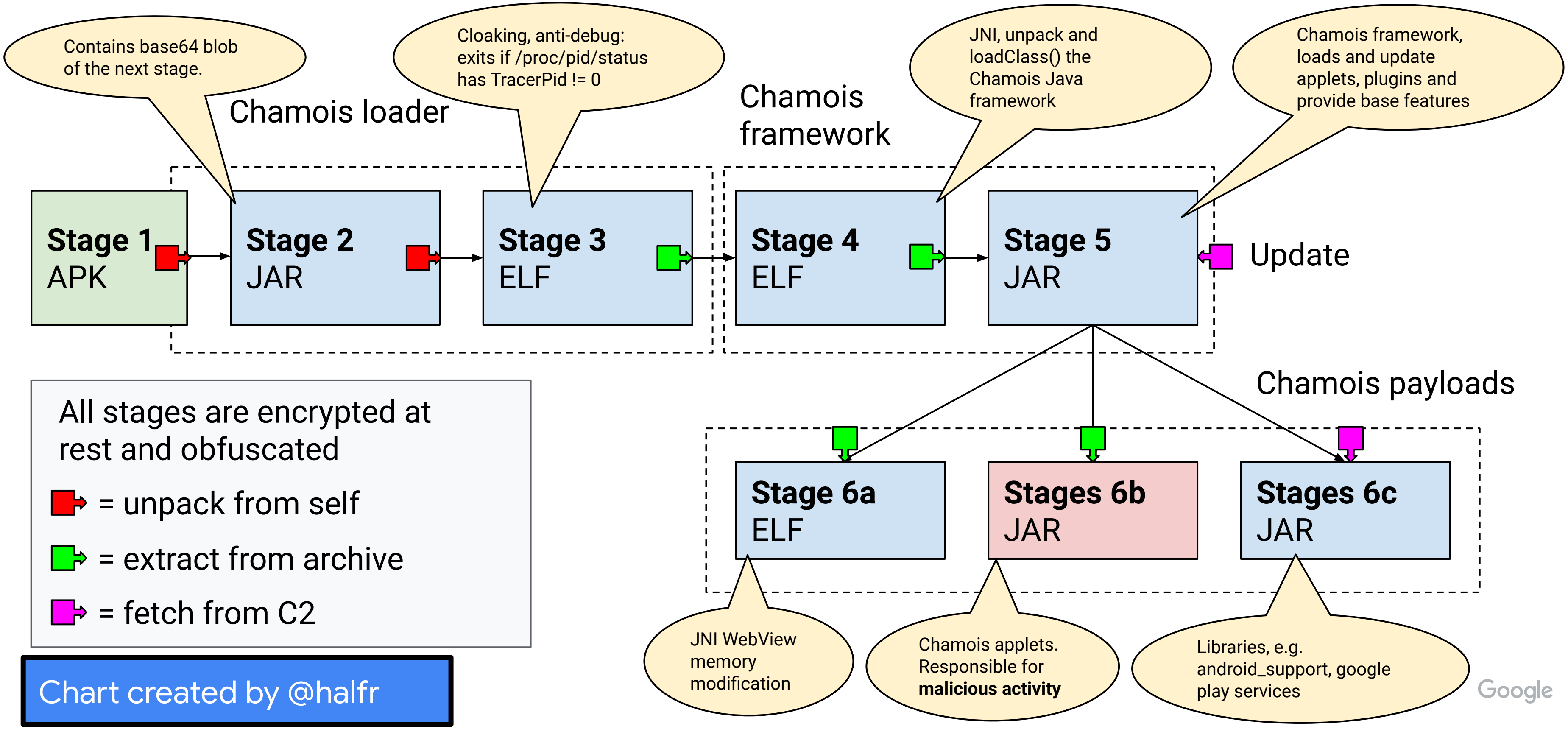
- V1: Aug 2016 - Mar 2017
  - Ad fraud payloads
  - Google Play fraud
- V2: Nov 2016 - Mar 2017
  - New premium SMS fraud payload
  - Google Play fraud
- V3: Nov 2017 - Aug 2018
  - Additional stages
  - Overall more sophisticated
  - Pre-installed & off-Google Play
- V4: Aug 2018 - Present(ish)
  - off-Google Play

This talk will focus predominantly on the latest two variants

Stages - Variants #1 & Variants #2



Stages - Variants #3 & #4



Contains base64 blob of the next stage.

Cloaking, anti-debug: exits if /proc/pid/status has TracerPid != 0

JNI, unpack and loadClass() the Chamois Java framework

Chamois framework, loads and update applets, plugins and provide base features

Chamois loader

Chamois framework

Stage 1 APK

Stage 2 JAR

Stage 3 ELF

Stage 4 ELF

Stage 5 JAR

Update

Chamois payloads

Stage 6a ELF

Stages 6b JAR

Stages 6c JAR

JNI WebView memory modification

Chamois applets. Responsible for **malicious activity**

Libraries, e.g. android_support, google play services

All stages are encrypted at rest and obfuscated

🟥 = unpack from self

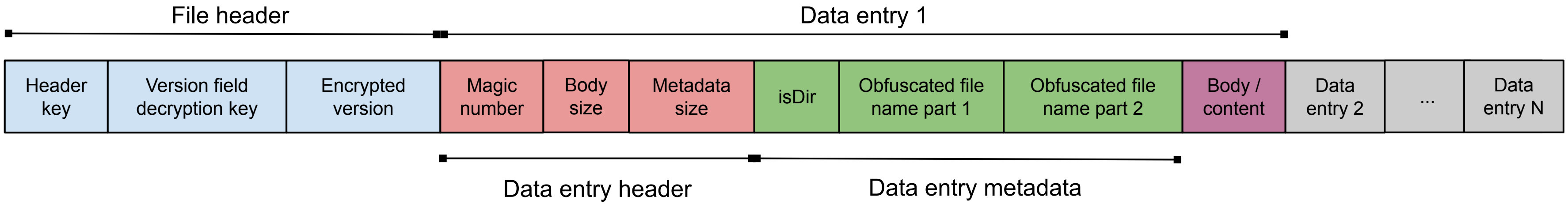
🟩 = extract from archive

🟪 = fetch from C2

Chart created by @halfr

Custom Archive Format

- Usage: similar to a ZIP containing JARs
- Supports directories & files
- Contains code packages, configuration and other support files
- Encryption: XXTEA, key material in the archive and in the app
- Used by multiple components: main framework and payloads



File header			Data entry 1									
Header key	Version field decryption key	Encrypted version	Magic number	Body size	Metadata size	isDir	Obfuscated file name part 1	Obfuscated file name part 2	Body / content	Data entry 2	...	Data entry N
			Data entry header			Data entry metadata						

Anti-detection techniques

- Stages 1 & 2 - randomized class names & file names for each new class name
- Stage 3 - ELF library containing sophisticated anti-analysis features (WeddingCake)
  - In-place decryption
  - Anti-reverse engineering
  - Anti-emulation
    - 37 system property checks
    - CPU architecture
    - Xposed and Monkey checks
  - For more information, see:
    - Blackhat USA 2018: “Unpacking the Packed Unpacker” video
    - VirusBulletin 2018: “Unpacking the Packed Unpacker” paper

Payloads

Mobile payment solution

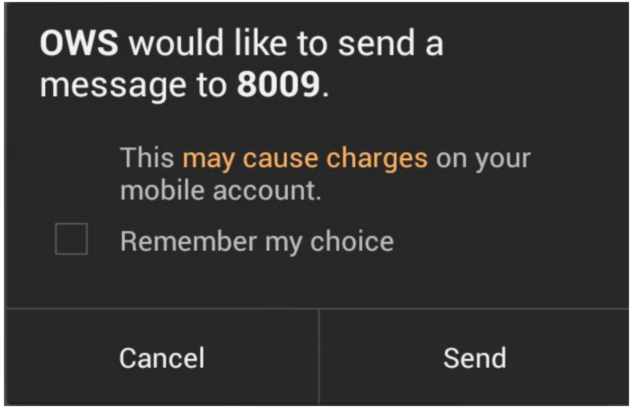
- Card payment
- SMS payment
- Mobile payment
- WAP payment

Malicious

- Ad fraud
  - Automated browsing
  - Click injection
  - Deceptive overlays
- App installs
- Traffic pumping
- Sends premium SMS

Premium SMS payload

- Apps must have the permission to send SMS
  - Chamois apps have it because they are phone-related
- Android platform asks the user to confirm send
  - Chamois use root access to enable internal permission flag and bypass the dialog
- No root, but accessibility services?
  - Use it to automatically tap "Send"



OWS would like to send a message to **8009**.

This may cause charges on your mobile account.

☐ Remember my choice

Testing Infrastructure

- Iterating on malware loader obfuscation to defeat existing rules
- Staging and production servers
- Multiple feature flags to control infected population behavior
- Progressive rollout of C2 configuration based on querying countries
- Using mobile analytics services and logging

Network infrastructure

- 10+ API C2 domains
- 20+ module-specific C2 domains
- 150+ domains for ad fraud activity
- Deployed on large cloud providers
- Automated cloud deployment
  - HTTPS with Let's Encrypt

But what if you’re in China?

```
public boolean isPushEnable() {
  if (SoftwareInfo.isChina()) {
     return false;
  }
  return read("Push", "enable", false);
}
public boolean isAdEnable() {
  if (SoftwareInfo.isChina()) {
     return false;
  }
  return read("AD", "enable", false);
}
public boolean isAdwebEnable() {
  if (SoftwareInfo.isChina()) {
     return false;
  }
  return read("ADWEB", "enable", false);
}
```

```
public boolean isAd2Enable() {
  if (SoftwareInfo.isChina()) {
     return false;
  }
  return read("AD2", "enable", false);
}
public boolean isSatelliteEnable() {
  if (SoftwareInfo.isChina()) {
     return false;
  }
  return read("Sate", "enable", false);
}
public boolean isGbRunnerEnable() {
  if (SoftwareInfo.isChina()) {
     return false;
  }
  return read("gbRunner", "enable", false);
}
```

Distribution

- Pre-installed
  - Convinced ODM and OEMs to include the SDK by advertising as a “mobile payment” solution
- Distributed to developers as a static SDK
- Sideloaded
  - Downloaded by apps as “plugins”
  - Distributed by other harmful downloader families

EagerFonts

- Fonts application included in SOC platform from 3P developer
- Included an advertising SDK that used dynamic code loading (DCL) to download from a 3P server and run “plugins” in the app context
- Plugins known malicious trojans:
  - Chamois - Backdoor
  - Snowfox - Trojan and Click fraud
  - And others.
- Affected 250+ OEMs across 1k+ different devices

- SOC Platform immediately pulled app, contacted their customers, and created a plan to prevent this issue in the future.

Fighting Chamois

Eradication Efforts

OEM Outreach	Stem the supply and distribution.
Google Play Protect	Protect users and block existing infections.
Ad Fraud Defenses	Prevent ability to monetize.

OEM Outreach

- Detected that some devices had Chamois pre-installed
- Initiated OEM Remediation process for devices in wild
  - 1) Alert OEM’s to presence on their devices
  - 2) Require OTAs to remediate
  - 3) OEM’s do post-mortem to determine how issued ended up on device
  - 4) OEM’s create plan for how they will prevent in the future
- Through certification program, test all potential new OEM builds for Chamois prior to approval and launch to users.

Google Play Protect

- Many types of automated detections
  - Signature based
  - Behavioral based
  - Network behaviors
  - Code similarity
  - Machine learning models
- More severe enforcement

Why was it hard?

Sophisticated Actor

- Industry presence/resources
  - Offers “monetization sdk” to OEM’s and ODM’s and references other entities
  - Using large cloud services
- Good engineering and release processes
- Sophisticated technical solutions
- Mature infrastructure

Stealthy

- Anti-analysis in depth:
  - Data encrypted at rest and deleted after load if dropped decrypted
  - Malicious payloads dynamically downloaded
  - Network traffic asymmetrically encrypted
- Anti-debugging in depth:
  - Network certificate pinning
  - Application certificate pinning
  - Anti-debugging at each stage
- Progressive rollout of payloads

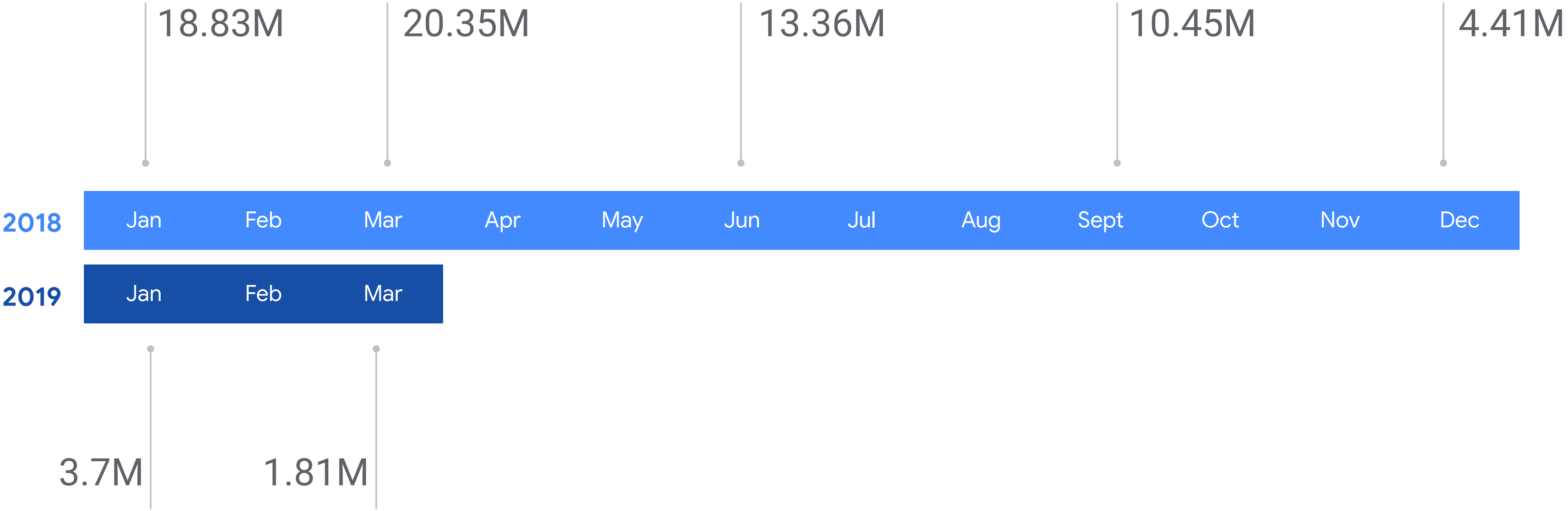
Rapid Response to Our Enforcements

- In response to enabling new detections, we often saw new samples that were trying to test the detections.
  - Moving bytes around, changing file, class, and string naming patterns
  - Removing some stages
  - New domains
- Fingerprinted Google's automated analysis environment

Chamois: Controlled.

By the numbers

Number of devices in the previous 28 days that had an active Chamois application



By the numbers: March 2018 until March 2019

48% increase in samples!

March 2018

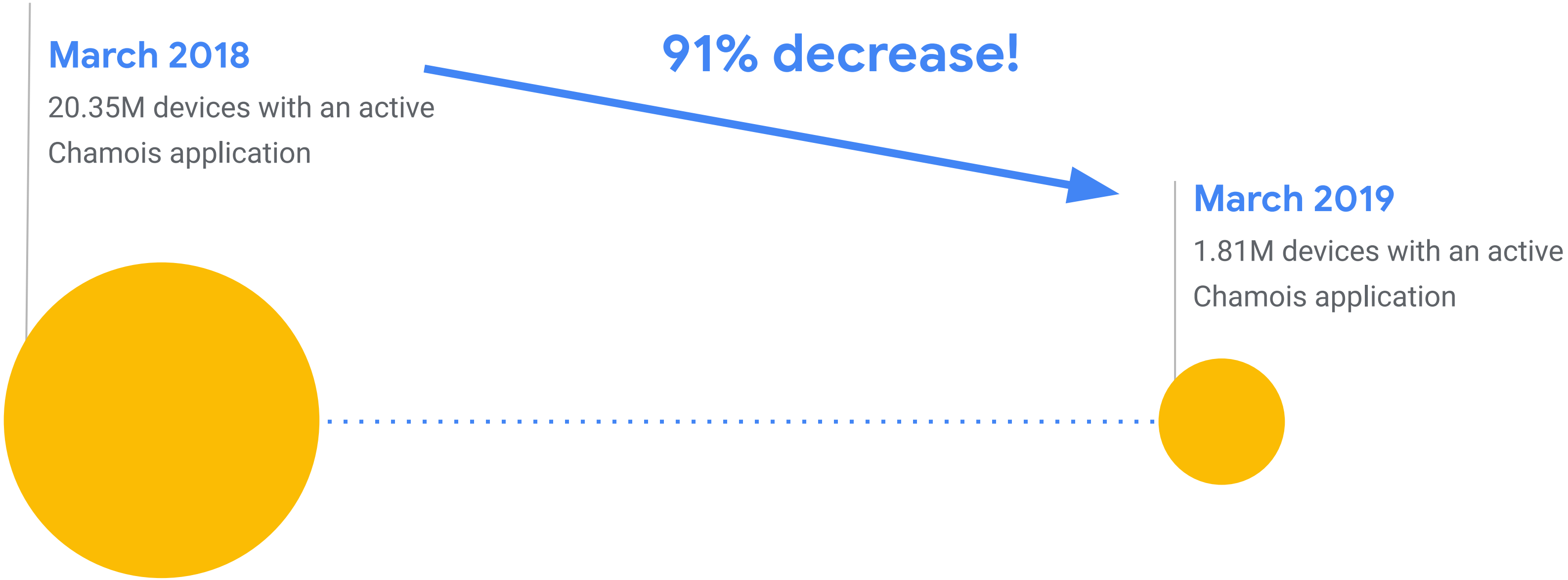
14k distinct Chamois samples

March 2019

27k+ distinct Chamois samples

12.8k new samples released!

By the numbers: March 2018 until March 2019



March 2018

20.35M devices with an active Chamois application

91% decrease!

March 2019

1.81M devices with an active Chamois application

The biggest botnet you never heard about.

Thank You

@maddiestone

Samples

54eaa874bbefbe78cb980e9cd90c9ec62eed0a4b73f738ded5c6d5b640e43869
ae9a81c9f8ce7c04021e13a6faf5976a3adab6d365d9db54a3f6a7ecffcfb2f2
e8e1bc048ef123a9757a9b27d1bf53c092352a26bdbf9fbdc10109415b5cadac
44c8cff9dc44e43b22cfc480b8397a2db6b9e271a655731565149608 56e27bb8