

Unconventional Cyber Attacks: Facts - Not Fakes



NATO Cyber Operations
Seminar
Greece, December 2020

Dimitris Gritzalis, George Stergiopoulos

INFOSEC Laboratory, Dept. of Informatics
Athens University of Economics & Business, Greece

Introduction

An unconventional attack

Dropping malware modules to multiple smartphones, over the air and from some meters away, using: (a) the sound medium, and (b) an algorithm (similar to Shazam's audio fingerprinting).

Trojan droppers

- Carrier or delivery vehicles for the payload to be dropped on victim machines.
- Open a way for attack (*download and install core malicious modules*).
- Among the top 5 worst malware threats; especially for Android.
- Network channels are hardened to prevent droppers from delivering their payloads.
 - This paves the way for alternative routes of infiltration.



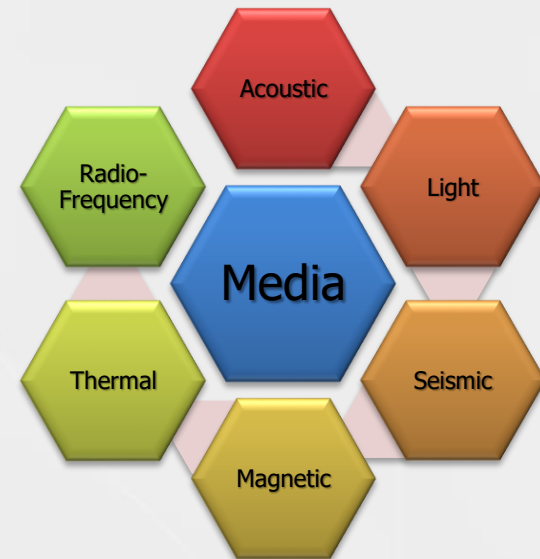
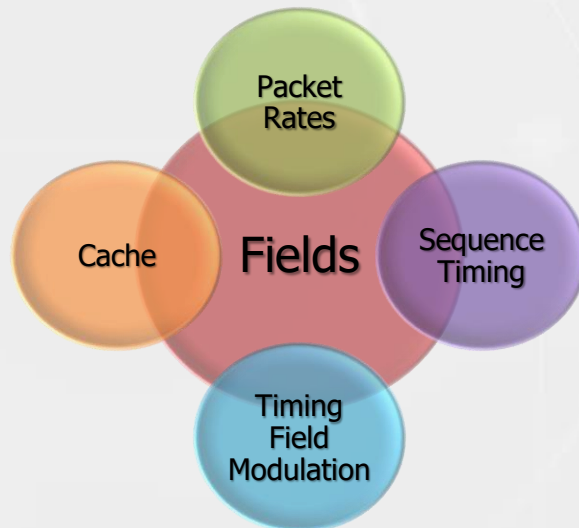
Unconventional Communication Channels

Covert Channels

- Mechanisms not designed for communication.
- Can be abused to transfer information objects between processes not supposed to communicate.

Such attacks:

- Utilize several fields.
- Exploit several media.



Related Attacks

Bad-Bios (2013)

BIOS level malware which communicates with ultrasonic sound between air-gapped laptops.

AirHopper (2014)

Exfiltrate data from an isolated computer to a nearby mobile phone, using FM frequency signals.

BitWhisper (2015)

Covert signaling between air-gapped computers using thermal manipulations.

GSMem (2015)

Exfiltrate data from air-gapped computers over cellular frequencies.

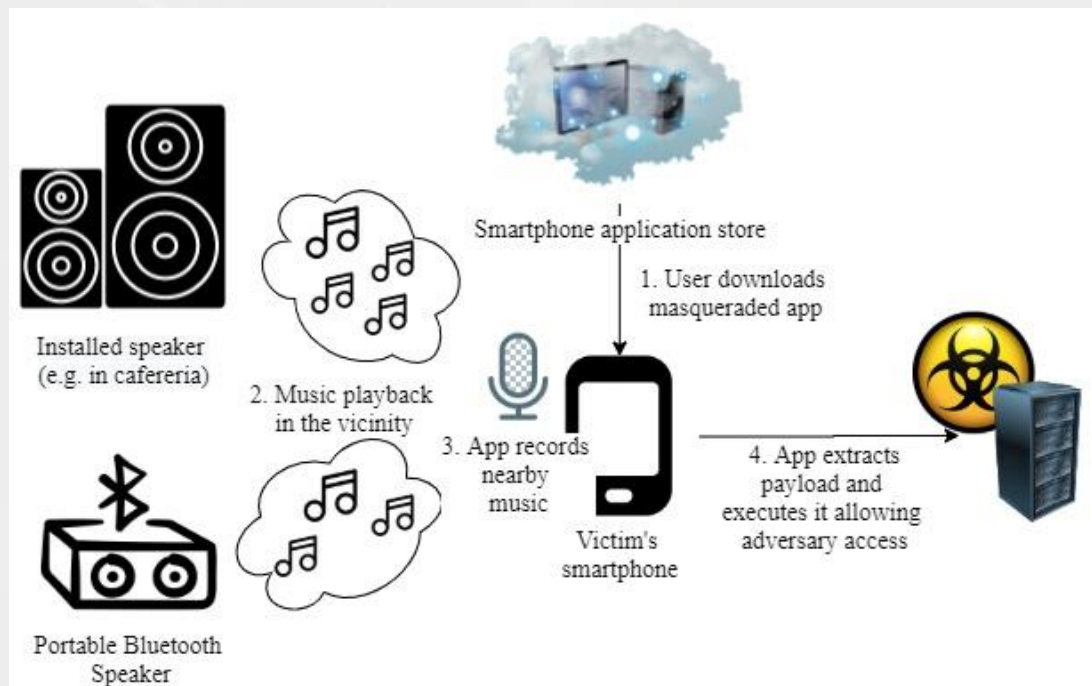
C³APSULe (2020)

Leak data across FPGAs through voltage-dependent channels.



Dropping malware through sound injection

- A malware payload is concealed within music's inaudible frequencies.
- A dropper software masquerades as a smartphone app that needs microphone access.
- Injected music is played back near the smartphone with dropper app installed.
- Attacker takes control of an Android device and acts maliciously using Meterpreter commands (*e.g. take photo, display running process, search for a file, record sound, etc.*).



Attack Model

Necessary implementations to drop payloads via music

Malware music injector (*MATLAB script*)

- Executes an STDFT algorithm to convert music to frequency tables and back.
- Embeds payload data as peaks of magnitude inside inaudible audio frequencies.

Dropper software app (*smartphone application*)

- Records music as an audio file via the microphone.
- Generates audio fingerprints of the file by mapping peak frequencies using STDFT.
- Uses this mapping to locate and extract hidden malicious symbols.
- Converts them to shellcode payload based on a hardcoded mapping.

Short-Time Discrete Fourier Transform (STDFT)

- Yields frequency information at different time instances.
- Spectrogram: intensity plot of STFT magnitude over time (*horizontal axis: time, vertical axis: frequency*).



Attack Model – Alphabet Definition

Shellcode

- Small series of bytes that represent executable machine-language code.

Injection range: ~15.0-19.5 KHz

- In principle, the human ear can hear approximately between 20Hz and ~18KHz.
- In fact, we do not hear sounds above 14KHz, even lower when noise present.

Payload: Encoded in base64 characters

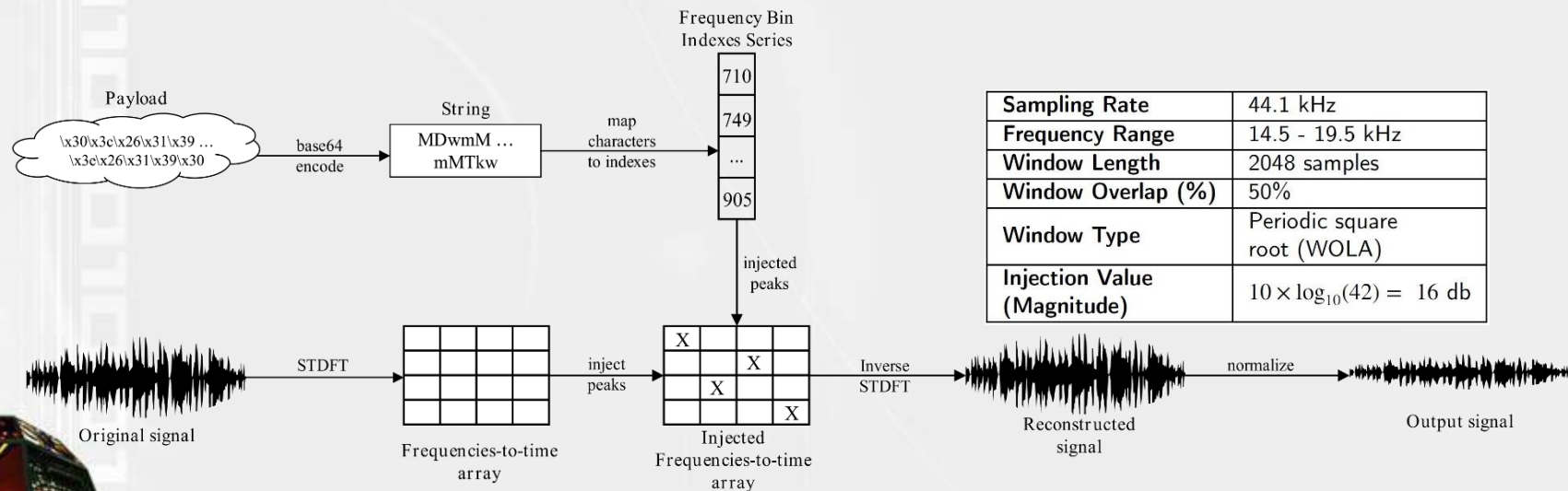
- Every base64 symbol is assigned to a specific frequency bin.
- 66 frequency bins in total, i.e.:
 - 64 for all base64 characters
 - 2 markers for beginning and ending of a full malware repetition

Base64 Character	Assigned Frequency-Bin Indexes
start	710
A	713
B	716
...	...
+	899
/	902
end	905



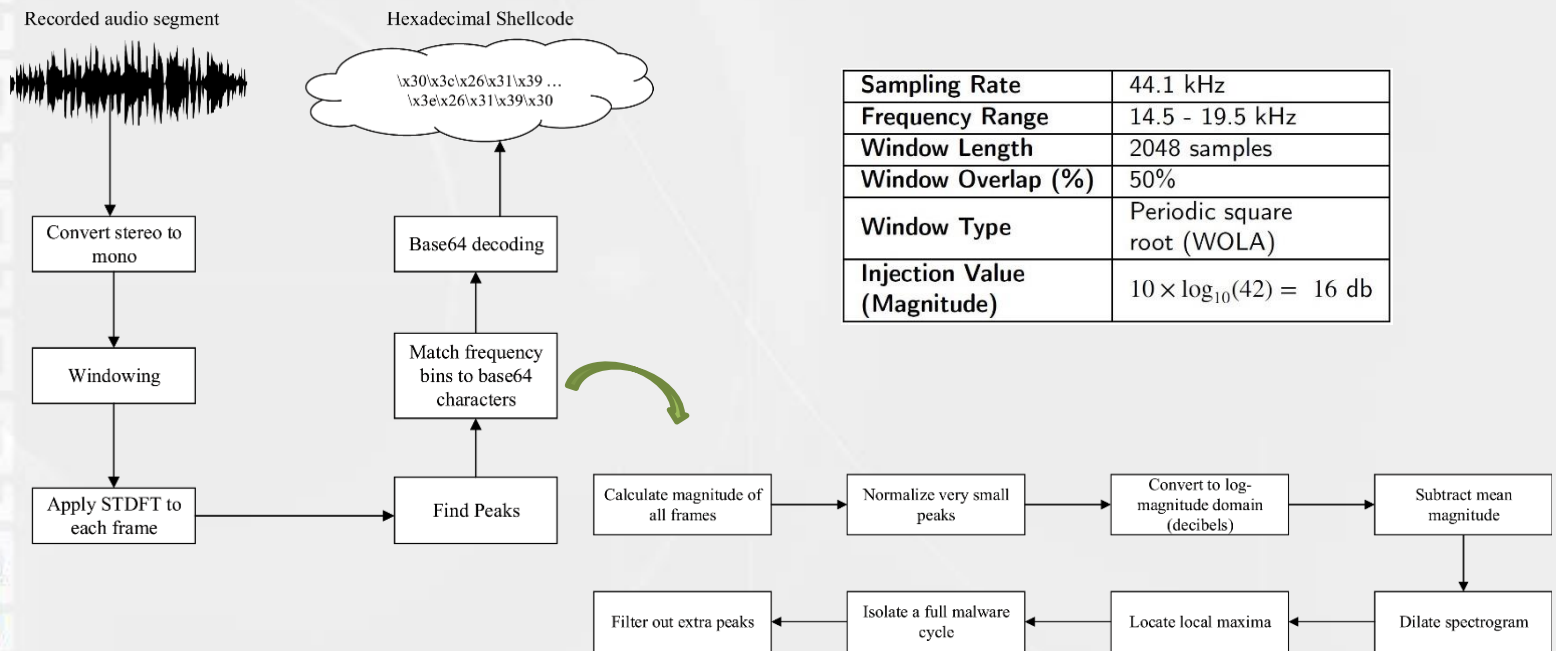
Attack Model – Injection Process

1. MATLAB script: Takes as input a shellcode payload, embeds it into a ".wav" audio file.
 - Assigns 16.2Db power to mark a particular frequency as an inserted peak.
 - Higher values produce audible distortions.
 - Lower values have dramatic impact on attack's effective distance.



Attack Model – The Dropper

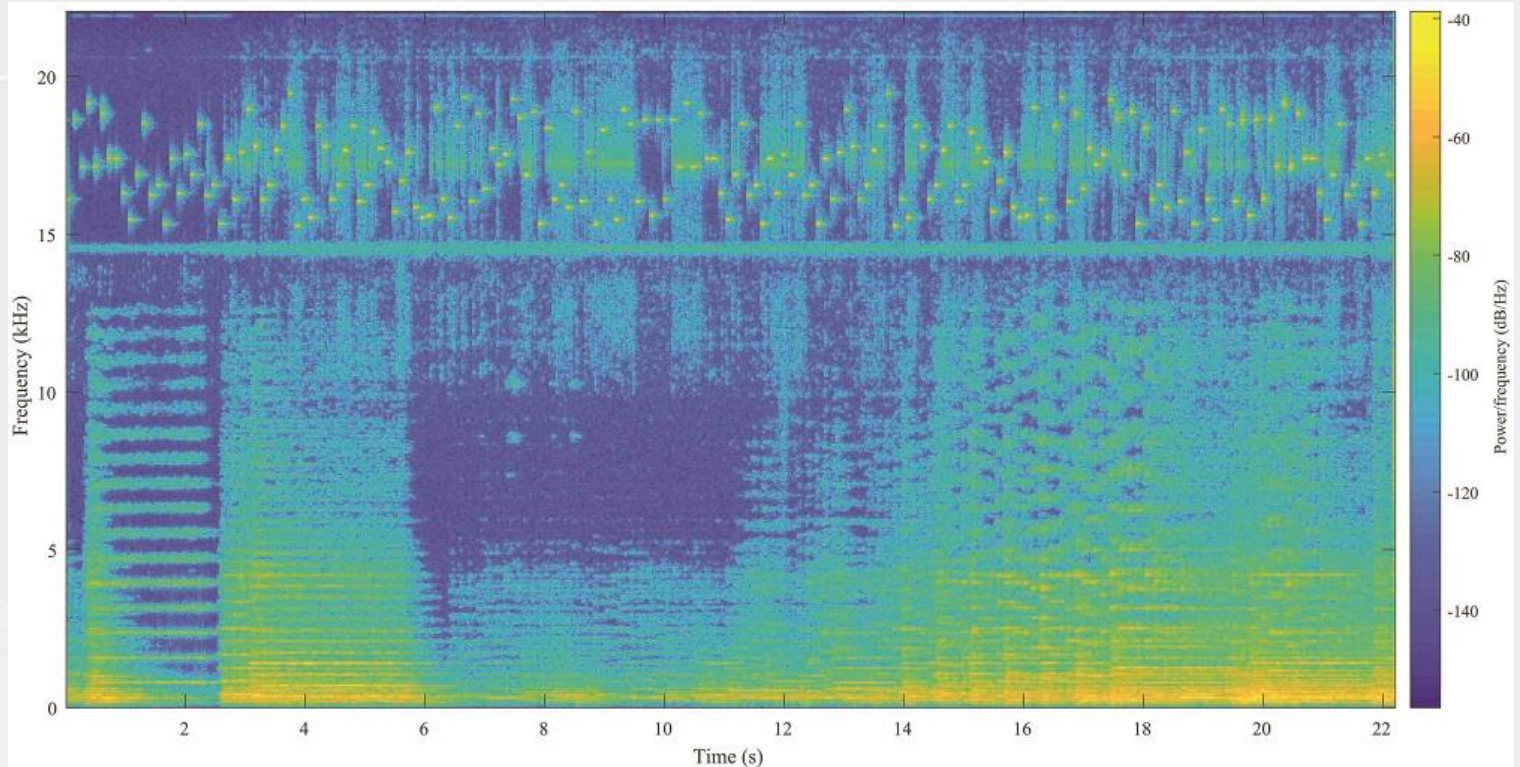
2. The dropper app is installed inside the victim's smartphone.
 - I. Victim grants permissions for microphone access.
 - II. For random, limited amounts of time, dropper app records music played in the vicinity.
 - III. If dropper detects symbols of malicious data, starts recording and extract malware.



Attack Experimentation

A 65-byte Meterpreter reverse TCP payload was injected to 22 different songs.

- Use of different music genres and languages (*rock, pop, folk, English, Greek etc.*)



Attack Experimentation - Settings

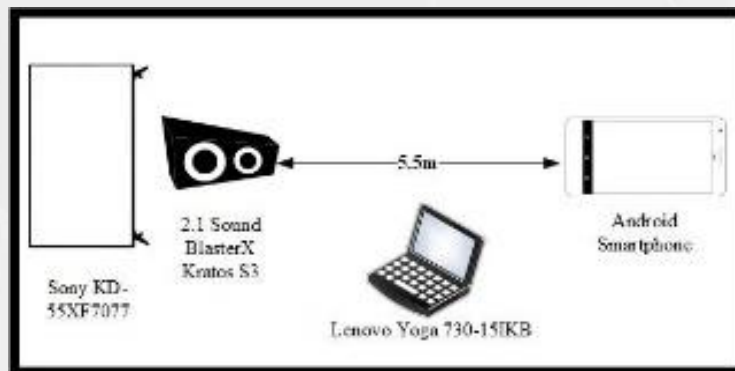
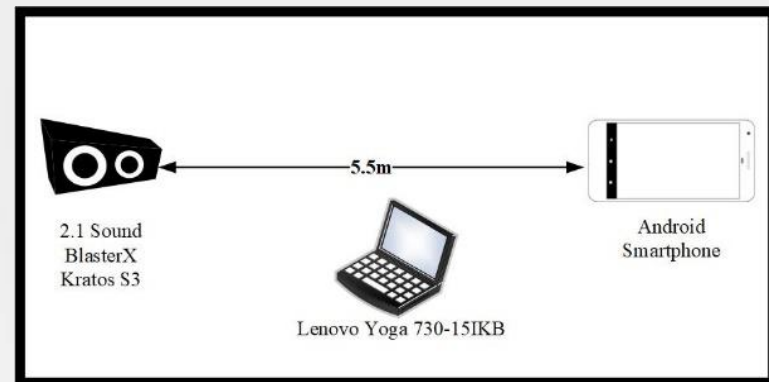
- Each experiment was performed with random recordings on each song.
- Music injector and dropper smartphone app share a hardcoded table that maps each base64 character to a high frequency bin.
- Recording
 - *Minimum recording duration* : Payload's length - first symbol
 - *Worst-case scenario* : Recording begins immediately after a starting mark.
 - $2 \times \text{payload's length} - \text{first symbol} = 20 \text{ seconds}$
- Equipment
 - Inexpensive non-professional 120\$ off-the-shelf speakers.
 - 3 different smartphone manufacturers (various versions of Android OS).



Attack Experimentation - Environments

Quiet Environments

- Only ambient music being played back.
- Speaker Volume: 8-52% of total power (4-26 DBm).
- Recording Distance: 0.5-10m.

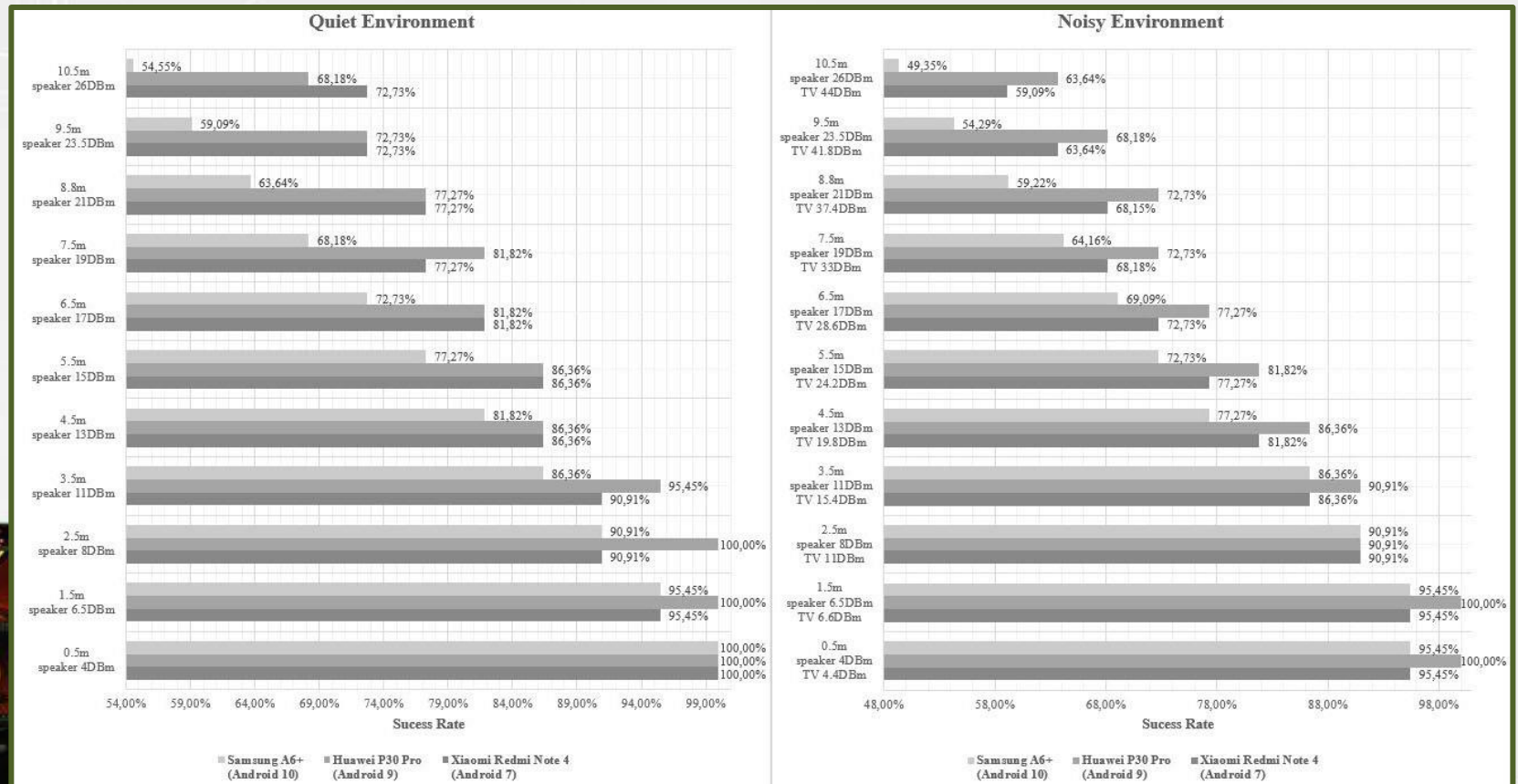


Noisy Environments

- Random background noise through a Smart TV and people chatter.
- TV Volume: 10-100% of total power (4.4-44 DBm).
- Recording Distance: 0.5-10m.

Experimental Results (1/2)

- Higher success rates of malware transmission in total silence room.
 - Overall performance: 82.78% quiet environments and 78.12% noisy ones.
- Average loss on unsuccessful attacks: 2 base64 characters.
 - 2.68% of payload's length*



Experimental Results (2/2)

Comparative analysis of **Smartphones** (*average performance*)

Distance	Huawei P30 Pro <i>(brand new)</i>		Xiaomi Redmi Note 4 <i>(moderately used)</i>		Samsung A6+ <i>(heavily used)</i>	
	<i>Quiet Environment</i>	<i>Noisy Environment</i>	<i>Quiet Environment</i>	<i>Noisy Environment</i>	<i>Quiet Environment</i>	<i>Noisy Environment</i>
≤2.5m	100%	96.97%	95.45%	93.94%	95.45%	93.94%
≤6.5m	87.5%	84.09%	86.36%	79.55%	79.55%	76.36%
≤10.5m	75%	69.32%	75%	64.77%	61.37%	56.76%

Comparative analysis of **Environments**

Distance	Quiet Environments		Noisy Environments	
	<i>Minimum Success Rate</i>	<i>Maximum Success Rate</i>	<i>Minimum Success Rate</i>	<i>Maximum Success Rate</i>
≤2.5m	90.91%	100%	90.91%	100%
≤6.5m	72.73%	81.82%	69.09%	77.27%
≤10.5m	54.55%	81.82%	49.35%	72.73%



Mitigation Techniques

Audio Source Level

- High-peak compression and modulation of sound-based channel frequencies.
- Filtering and compression of audio files to distort any existing ultrasonic frequencies as they pass through the amplifier.

Smartphone Device Level

- Programmers do not use unnecessary background activities in application.
- Users decide about permissions for sensor access when installing an application.
- Smartphone manufacturers set microphone sensitivity $<15\text{kHz}$, to avoid capturing inaudible frequencies.



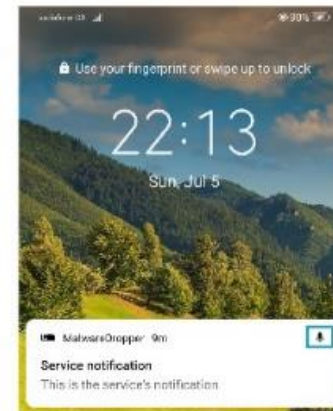
Limitations

Open issues

- Google limited access to device sensors for background apps:
 - Users notified when an app is accessing a sensor.
 - Android 9 Pie onwards.
- Viability of the attack is affected by:
 - Hardware-specific factors:
 - I. Microphone quality/sensitivity among different manufacturers.
 - II. Poor device handling.
 - III. Extended mobile use.
 - Surrounding environment related factors:
 - I. Background noise.
 - II. Low DB volume on the speaker.

Newer Android versions

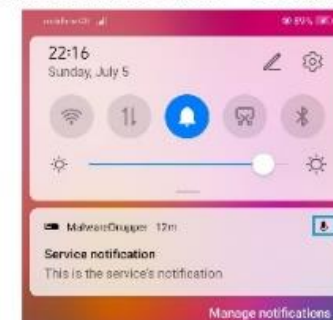
Locked Screen - Recording



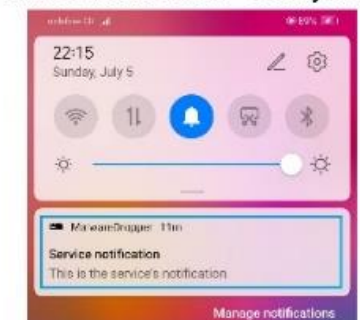
Locked Screen - Analyzing



Notification Panel - Recording



Notification Panel - Analyzing



Conclusions

- An over-the-air payload dropping/injection technique on mobile devices without corrupting original audio signal was established.
 - No professional equipment is needed.
- Only 20 sec recording is needed for typical Meterpreter payloads.
- Most effective distance: ~5m away from the music source.
 - Still viable up to 8m.



References

1. Carrara B., Adams C., "On acoustic covert channels between air-gapped systems", *Proc. of the International Symposium on Foundations and Practice of Security*, pp. 3-16, Springer, 2014.
2. Deshotels L., "Inaudible sound as a covert channel in mobile devices", *Proc. of the 8th USENIX Workshop on Offensive Technologies*, 2014.
3. Gritzalis D., "Embedding privacy in IT applications development", *Information Management & Computer Security*, Vol. 12, No. 1, pp. 8-26, 2004.
4. Guri M., Kedma G., Kachlon A., Elovici Y., "AirHopper: Bridging the air-gap between isolated networks and mobile phones using radio frequencies", *Proc. of the 9th International Conference on Malicious and Unwanted Software*, pp. 58-67, IEEE, 2014.
5. Iliadis J., Gritzalis D., Spinellis D., Preneel B., Katsikas S., "Evaluating certificate status information mechanisms", in *Proc. of the 7th ACM Computer and Communications Security Conference*, pp. 1-9, ACM Press, 2000.
6. Mylonas A., Dritsas S., Tsoumas B., Gritzalis D., "Smartphone security evaluation: The malware attack case", *Proc. of the International Conference on Security and Cryptography*, pp. 25-36, 2011.
7. Mylonas A., Gritzalis D., Tsoumas B., Apostolopoulos T., "A qualitative metrics vector for the awareness of smartphone security users", *Proc. of the International Conference on Trust, Privacy and Security in Digital Business*, pp. 173-184, Springer, 2013.
8. Mylonas A., Kastania A., Gritzalis D., "Delegate the smartphone user? Security awareness in smartphone platforms", *Computers & Security*, Vol. 34, pp. 47-66, 2013.
9. Mylonas A., Meletiadiis V., Mitrou L., Gritzalis D., "Smartphone sensor data as digital evidence", *Computers & Security*, Vol. 38, pp. 51-75, 2013.
10. Mylonas A., Theoharidou M., Gritzalis D., "Assessing privacy risks in Android: A user-centric approach", *Proc. of the International Workshop on Risk Assessment and Risk-driven Testing*, pp. 21-37, Springer, 2013.
11. Rasmussen K., Giechaskiel I., Szefer J., "CAPSULE: Cross-FPGA covert-channel attacks through power supply unit leakage", *Proc. of the IEEE Symposium on Security and Privacy*, Vol. 1, No. 2020, IEEE, 2015.
12. Salonikias S., Mavridis I., Gritzalis D., "Access control issues in utilizing Fog Computing for Transportation Infrastructures", *Proc. of the 10th International Conference on Critical Infrastructures Security*, pp. 1-12, Springer, 2015.
13. Virvilis N., Gritzalis D., "Trusted Computing vs. Advanced Persistent Threats: Can a defender win this game?", *Proc. of 10th IEEE International Conference on Autonomic and Trusted Computing*, pp. 396-403, IEEE Press, 2013.