The Q&A section of the webinar had some interesting questions about protection of digital forensic tools against reverse engineering and about vulnerabilities in digital forensic tools.

So, here are additional comments.

1. Many proprietary digital forensic tools are distributed in the obfuscated form. Sometimes software protection systems (like Oreans Technologies Themida) are used.

This makes reverse engineering to locate weak spots for further validation hard or nearly impossible (while this does not affect black-box tests, such tests have significant limitations – this was discussed before). Similarly, discovering vulnerabilities (already known and previously unknown) in protected software becomes harder.

2. Many digital forensic tools bundle third-party libraries to provide “stable” functions across different environments. If these libraries are outdated, they can become a source of vulnerabilities.

The number of bundled third-party libraries and tools is high for mobile forensic tools.

3. Some mobile forensic tools deliberately bundle outdated libraries and tools to support old mobile devices.

For example, some Android phones are incompatible with current versions of the ADB tool, so an outdated version of this tool can be packaged (along with a newer version for other Android phones) to run on the host computer.

A vendor can backport patches to fix known vulnerabilities if the source code is available... But who knows? In one mobile forensic tool (which is not named here), a very old (from 2013) “vanilla” (i.e., not patched and not even recompiled with additional protections) version of the ADB tool is shipped along with newer versions.

This practice significantly increases the attack surface.

4. As with any software, users can run outdated versions to work around known (but not yet fixed) bugs. Vendors often suggest software downgrades as a workaround.

5. In general, vulnerabilities in digital forensic tools (and in linked general-purpose libraries and tools) can be divided into two categories:

- Not affecting digital forensic functions (at least directly).

  For example, DLL search order hijacking issues that result in local privilege escalation (e.g., an attacker having local or remote access to a guest account on the computer can gain administrator privileges). Another example is insecure permissions for files, directories, and registry keys that allow unprivileged users to do something that is not normally expected or allowed.

  Usually, such vulnerabilities cannot be exploited without prior access (remote or local) to the examiner's computer (but they can be used in exploit chains covering multiple vulnerabilities). Additionally, some vulnerable executables (e.g., susceptible to DLL

hijacking) can be extracted from digital forensic tools and used to covertly launch malicious code on other computers (a vulnerable executable, which is digitally signed and considered as known good by anti-malware products, can be tricked into running custom code – this allows attackers to bypass security features like application allowlisting).

- Directly affecting digital forensic functions.

  This includes code execution issues affecting the examiner's computer (or digital forensic devices like hardware write blockers and hardware forensic imagers). After successful exploitation, data stored on devices being examined (or acquired) can be deleted or altered (unless these devices are truly write-blocked and this protection cannot be turned off programmatically) or data stored on the examiner's computer (like disk images and reports) can be modified. If the examiner's computer is not air-gapped, a backdoor can be installed to enable remote access.

  For example, many live Linux distributions (including those having forensic capabilities) can execute code stored on an internal drive automatically (without any user input) during the boot (https://dfir.ru/2018/07/21/a-live-forensic-distribution-executing-malicious-code-from-a-suspect-drive/). Another example is a vulnerability found in Cellebrite UFED 4PC, which can be exploited during the acquisition of a mobile device (https://signal.org/blog/cellebrite-vulnerabilities/).

6. Usually, vulnerabilities in digital forensic tools are underrated.

A common point of view can be summarized as follows:

Theoretical scenarios and proof-of-concept exploits are not enough to render digital evidence inadmissible in a particular case. It must be demonstrated that a particular vulnerability has been successfully exploited in this case, leading to specific changes of data (or other consequences).

As with any software, digital forensic tools have bugs. Some of these bugs are vulnerabilities. And some of them are not yet disclosed (also known as zero-day). It is okay when vendors patch vulnerabilities in a timely manner. It is not okay when vendors ignore known vulnerabilities and do not fix the software (unless these vulnerabilities are in the news). The absence of legal consequences in a particular case (like rendering digital evidence inadmissible) is not an excuse to ignore security issues.

While examiners can reduce the risks (https://www.sans.org/blog/arbitrary-code-execution-on-examiner-systems-via-file-format-vulnerabilities/), it is important for vendors to reduce the attack surface, introduce mitigations (measures that make successful exploitation of vulnerabilities harder) and controls (like security sandboxing), and implement the vulnerability management processes.

6.1. Let us assume that an examiner's computer is infected with a malicious program that places illicit material into some devices being acquired.

According to one blog post (http://cyberlaw.stanford.edu/blog/2021/05/i-have-lot-say-about-signal%E2%80%99s-cellebrite-hack), a different tool can be used to demonstrate that no exploits spoiled the digital evidence (by performing another acquisition and comparing its results to the previous one, which was allegedly affected by an exploit).

This is not going to work in many scenarios, especially with mobile devices (because they cannot be truly write-blocked) – when original data is spoiled or faked (modified in some way during the first acquisition), comparing its copies is meaningless, because “spoiled or faked bytes” make their way into both copies (two tools are going to read the same overwritten blocks of data).

There are some other ways to verify the reliability of digital evidence:

- use different types of evidence (e.g., a witness testimony or a confession) to confirm that the data in question existed before the digital forensic tool was used (so, this data was not planted during or after the acquisition);

- if multiple devices have been seized and some of them were examined (or acquired) using a different tool, find references to the data in question, or copies of that data, on those devices (to prove that the data in question exists on other devices, which were not affected by a vulnerable tool);

- similarly, more sources of digital evidence, like warrant returns and cloud data, can be used to find references to the data in question (or copies of that data);

- provide evidence that a user knew about the data in question before the seizure of the device (e.g., a user viewed or edited that data; but such traces could be planted as well);

- provide evidence that the data in question was written to the device memory before the seizure (e.g., relevant files and database entries were not backdated or timestomped);

- examine the machine used to acquire the data and demonstrate that there were no traces of exploits and active malware (this is not going to work if the acquisition tool used is, or is a part of, a live distribution, because all relevant data was stored in the random-access memory, which is lost when the computer is powered off).

If no different evidence is available, no alternative sources of digital evidence (other devices, warrant returns, cloud data, etc.) are available (or they do not contain relevant information), and the examiner’s machine is unavailable (e.g., it was decommissioned or reimaged a long time ago, or it cannot be examined due to legal reasons), the only source left is the device itself. And such situations are not unusual.

In these situations, can we rely on examiners to identify backdated or timestomped files and database entries? Is it enough to prove that evidence was not spoiled or planted? If yes, why do we implement (and, in many situations, require the use of) additional measures like tamper-evident bags when dealing with physical access to evidence?

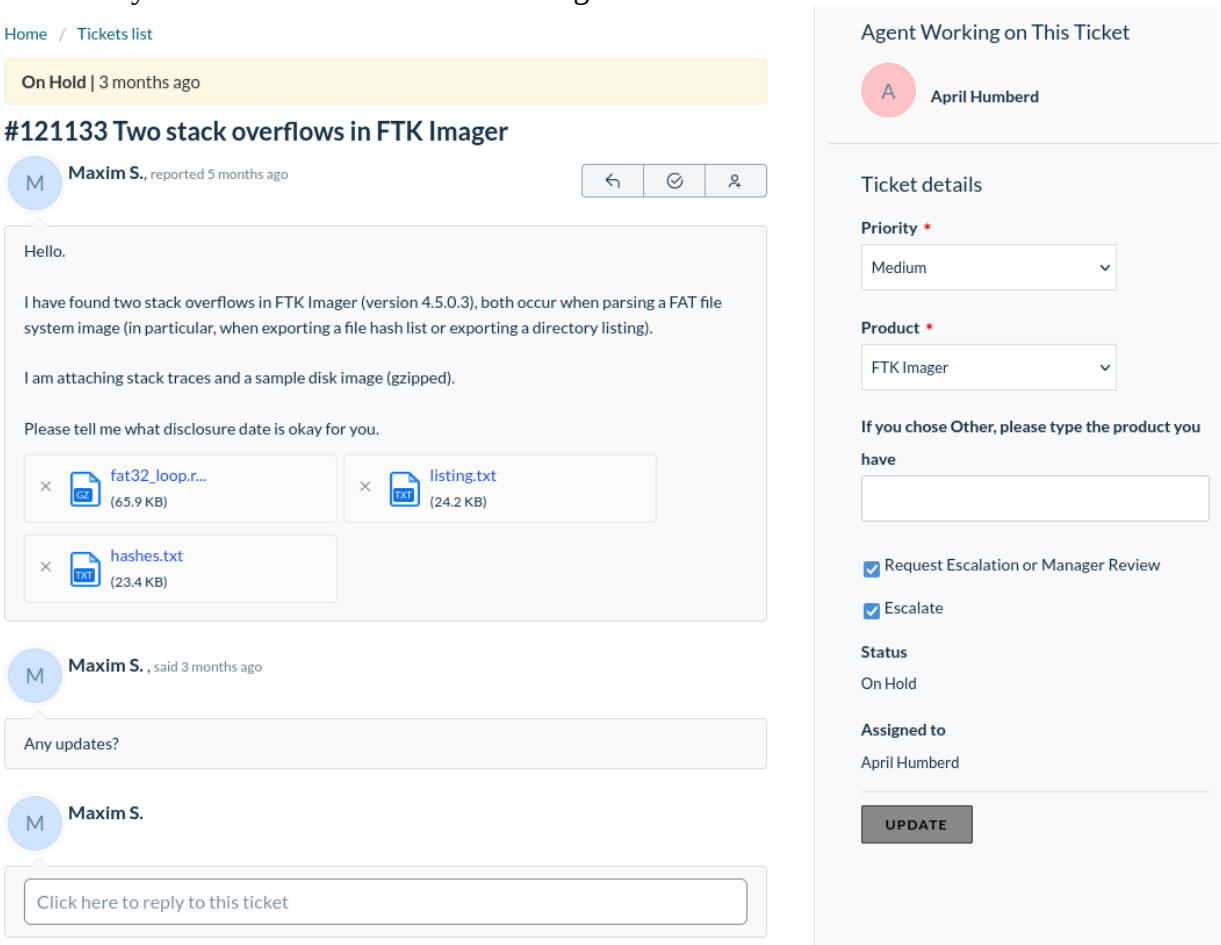
Or do we need additional protections here? If so, why do we rely on tools from vendors who do not care about security? Can we compare the *ability* of gaining unauthorized and undetected physical access to evidence (e.g., when no tamper-evident bags or similar packages are used to store and transfer evidence) with attaching evidence to a computer running a digital forensic tool having known vulnerabilities (or to a computer running a digital forensic tool that could be successfully exploited before) in terms of reliability? (We cannot overcome zero-day vulnerabilities, but we also have known ones, which can be around in digital forensic tools for years.)

7. Some vendors and maintainers of digital forensic tools occasionally ignore vulnerability reports.

Here are two recent examples:

- Multiple security issues with The Sleuth Kit: https://github.com/sleuthkit/sleuthkit/issues/2641.

- Two security issues with the Exterro FTK Imager:



(As of 2022-06-08, these stack overflows are not fixed.)

8. Vulnerabilities in digital forensic tools can affect other software products.

For example, a vendor can decide to reuse vulnerable file system parsing code taken from a digital forensic tool in an enterprise product (e.g., in an EDR tool). These new software products can be deployed in large-scale corporate networks, delivering vulnerabilities to their workstations and servers.