USB Flash Storage Threats and Risk Mitigation in an Air-Gapped Network Environment

George E.Pajari, CISSP, CISM

@OrangeHazMat
George.Pajari@HCIS.ca

This version of my paper is a final draft that still needs some work fixing the footnotes and references so I would appreciate your limiting the circulation until I can produce the final version. I also expect to add some material based on the questions and comments from my presentation at CanSecWest 2014.

If you would like to be notified of when that happens (expected by March 20th 2014), please send me email.

PLEASE please email me your comments and suggestions. Not for a moment do I think I have considered all aspects of this issue and I will warmly receive any and all feedback (even if you think the work is trivial, derivative, and unoriginal).

Thanks.

George Pajari
West Vancouver, BC
2014-03-12

USB Flash Storage Threats and Risk Mitigation in an Air-Gapped Network Environment

George Pajari, *CISSP, CISM*

***Abstract*—The sine qua non of information system security is the air gap – computer(s) completely disconnected from the outside world. This utmost level of security is frequently mandated for systems that process extremely sensitive information, control critical infrastructure, or hold the keys to the kingdom (metaphorically speaking).**

Recently, the use of air-gapped systems to store bitcoins has become popular amongst on-line bitcoin wallets and exchanges and is referred to as "cold storage".

But as no man is an island, few computer systems can operate usefully without exchanging information with systems that are connected, and usually the method of moving data across the air gap is USB removable media.

But therein lies a potential achilles heel – USB storage devices are vulnerable to a number of threats that can undermine the security provided by the gap.

This paper will provide a threat taxonomy for air-gapped systems and propose a number of methods to mitigate these threats.

***Keywords*—security, air gap, bitcoin, USB, threat taxonomy, vulnerability analysis**

I. Introduction

IT is a widely accepted fact that in order to provide the highest levels of security for computer systems, they must not be connected to the Internet.[1]

Harvard University's security policy for "Level 5 information" (extremely sensitive research information about individually identifiable people) requires that the computers be physically protected and not be connected to a network that extends outside of the room. [2]

Bitcoin exchanges boast that they keep most of their trust funds in "cold storage". From one such exchange's marketing material:

> "Cold Storage is the act of sending bitcoins to an offline wallet address. Access to withdraw these funds must be by a human being and with a computer that is never plugged into the internet. This guarantees that a hacker cannot steal the wallet through the internet. This is done with transaction signing and USB keys to transfer the signed transaction from the offline computer to the online Bitcoin network. Over $500,000USD was stolen from the [other exchanges] by hackers because they did not use cold storage. [We use] cold storage on approximately 80% of customer funds." [3]

This description of *cold storage* touches on the key problem with air-gapped (offline) computers — the need to move data across the gap without compromising the security benefit of having isolated the offline computer in the first place.

II. Do Air-Gapped Computers Really Exist?

> "The only truly secure system is one that is powered off, cast in a block of concrete, and sealed in a lead-lined room with armed guards - and even then I have my doubts."[2]

There is some debate about whether or not a completely air-gapped computer system exists. Eric Byres, CTO of Tofino Security (a company specialising in products to secure SCADA[3] networks) has written and spoken extensively on what he refers to as the myth of the air gap [5]. He correctly points out that in almost every situation in which the system is disconnected from the Internet or the corporate LAN, there is the electronic transmission of data, even if intermittent. It may be by connecting a notebook computer temporarily to the network, it may be by transferring data using removable media, but he argues that precious few so-called "air gaps" are true air gaps.

While his point is valid, it does not mean that the concept of the network air gap is passé. Perhaps what we need is a hierarchy of air gap "categories":

- Cat 0 is the true (if non-existent or mythical) air-gapped environment.
- Cat 1 is an air-gapped network where data only ever flows from the air-gapped systems to an external network using a data diode or removable media that ONLY ever travels in one direction.
- Cat 2 is an air-gapped network where data crosses the gap (in both directions) using removable media.

Note that a category 1 system can never be updated. Nothing, but nothing, can ever flow *to* the air-gapped systems, otherwise it is a category 2 network.

As renown security expert Bruce Schneier has written:

> "Air gaps might be conceptually simple, but they're hard to maintain in practice. The truth is that nobody wants a computer that never receives files from the internet and never sends files out into the internet. What they want is a computer that's not

G. Pajari is with HCIS Health Care Information Security, Inc. (www.hcis.ca)
Paper to presented at CanSecWest 12 March 2014; revised 12 March 2014.

[1]For example, "The most secure computers are those not connected to the Internet..." from Wikipedia [1]

[2]Gene Spafford [4]

[3]Supervisory Control and Data Acquisition (a.k.a. Industrial Control System)

> directly connected to the internet, albeit with some secure way of moving files on and off." [6]

The remainder of this paper attempts to address Schneier's challenge of providing "some secure way of moving files on and off".

A. A Note on Nomenclature

For simplicity and clarity this paper uses the following terms:

Infected
: refers to a computer system that harbours malware, whether or not the malware is a virus or non-propagating.

Protected
: refers to a computer system (or systems) that are to be isolated from the "rest of the world" by an air gap. These conputers are sometimes referred to as *offline*.

Inbound
: refers to data (or media) travelling across the air gap from an unprotected computer to a protected computer.

Outbound
: refers to data (or media) travelling across the air gap from a protected computer to an unprotected computer.

USB drive
: refers to a USB flash storage device.

III. Crossing the Air Gap

The mechanism we would propose was hinted at earlier: USB flash storage devices.

These are commonly used to cross the air gap because they are:

- available with large storage capacity (larger than CD-ROM or DVD-RW media);
- inexpensive and widely available
- durable and extremely portable
- reusable (further reducing their operational cost)
- fast (R/W rates typically far exceeding optical media)

Any removable storage can be a vector to introduce malware into a system or a vehicle to exfiltrate sensitive information form a system. USB storage devices, as we will see, are subject to all the same vulnerabilities of any removable storage as well as USB-specific threats.

These risks have been well publicised, including a report that a Russian astronaut brought a USB flash drive on board the International Space Station which infected systems on the spacecraft (see Gilbert [7]).

In order to develop methodologies for using USB storage devices for moving data to and from air-gapped systems without compromising the security of the air-gapped systems we must first develop a complete taxonomy of the threats that exist. Only then can we develop and analyze our methodologies to see if they provide adequate mitigation to the enumerated threats.

> "The fear of USBs post-Stuxnet has resulted in a number of even worse practises such as indiscriminate use of laptops and serial comms to move important log or process data off of the plant floor. When I see things like that, I think the cure is often worse than the disease when it comes to security."[4]

IV. Threat Taxonomy

At the top level of our proposed USB threat taxonomy we divide threats into three broad categories:

Malware on (USB) Removable Media
: All threats carried by a removable storage device to an air-gapped system.

Data Exfiltration via (USB) Removable Media
: All threats that attempt to exfiltrate sensitive information from an air-gapped system to the outside world unbeknowst to the operator (i.e. this category does not cover insider threats, instead an infected air-gapped system attempting to use removable media to extract information from the air-gapped system in addition to the information the operator is intending on moving).

Trojan USB
: All threats that involve a USB device that can change its logical appearance (either by hiding/revealing storage or changing the device type, for example, from storage to keyboard (see section below).

For completeness we include non-USB threats to air-gapped computers. We include these, not because we intend to address them in this paper, rather to remind the reader that the USB threats are not the only ones that need to be addressed in any comprehensive risk analysis of air-gapped computer systems.

Physical Threats
: All threats that involve physical access to the air-gapped systems.

Insider Threats
: All threats that involve trusted personnel violating policies or procedures either maliciously or unknowingly.

Radio Frequency Emanations
: When any electronic device operates, it generates radio frequency signals that can be picked up some distance from the device if the device is not properly shielded. It has been known since the 1940s that information printed by a computer could be eavesdropped at a distance of tens of metres. The US and other governments have standards for shielding equipment to reduce the ability to obtain information from computers and communications systems at a distance (see, for example, Tempest [8]).

WiFi and Bluetooth
: Many computers have some form of integrated wireless communications device supporting protocols

[4] Eric Byres, private email, with permission.

such as WiFi or Bluetooth. Obviously if an air-gapped computer becomes infected with malware, the existence of integrated wireless communications devices could provide a method for the malware to exfiltrate information.

Sound devices

Recent research (Hanspach [9]) has demonstrated that a malware infected computer can use the sound card to communicate with other infected computers across the air gap. There have also been reports (Dragos Ruiu's badBIOS [10]) that an air-gapped computer can be infected by a USB flash drive and then use the sound card to exfiltrate data.

NSA devices

The Snowden disclosures have revealed that the NSA has developed devices to permit surreptious access to air-gapped computers. These devices take the form of small circuit boards installed on systems intercepted between the manufacturer and end-user, or miniature transceivers embedded in USB cables (see Sanger et al in the New York Times [11]).

Note that the material released by Snowden is roughly five years old so we may assume that the techniques the NSA has developed for crossing the air gap are now much more sophisticated.

Those who feel that their threat landscape includes agencies such as the NSA (and CSEC in Canada) may wish to read the Wikipedia entry on Faraday cages [12].

V. Risk Analysis

As with all threat scenarios, one must conduct a risk analysis to determine which threats are worth mitigating. If your air-gapped network is storing bitcoins, your risk analysis will emphasize certain threats.[5]

If, on the other hand, you are collaborating with Edward Snowden, your risk analysis might put greater emphasis on threat actors that might not be worth considering in the bitcoin scenario.

VI. Malware

It is obvious that along with the files that are intended to be moved to the air-gapped system, malware can also be carried on the USB. Some of the possibilities include:

- the system that copied the data to the USB device was itself compromised (infected) and contaminated the USB storage device; and
- the USB device was contaminated prior to coming into the possession of the user.

Examples of the latter risk include contaminated (infected) USB drives:

- dropped in the parking lot to be picked up by staff (see below);
- placed in trade show give-away fish bowls;
- put back on the shelf at a retail outlet; and
- created accidentally during the production of a software distribution [6]

It ought to be noted that there are a number of possible ways malware on a USB storage device can infect the computer the device is inserted into other than *AutoRun* and *AutoPlay*. When a USB drive is inserted the operating system scans the device to determine its type, and if a storage device, attempts to mount the file system. Carefully constructed abnormal file systems have in the past exploited operating system defects with the potential to propagate malware even on systems that have disabled *AutoRun* and *AutoPlay* (e.g. Microsoft MS13-027 [13] and CVE-2010-2568). Social engineering can also be very effective in getting a user to open a file (i.e. by clever file naming such as "CONFIDENTIAL Payroll Data").

Jon Larimer has written an excellent survey of these vulnerabilities [14].

VII. Found USB Drives

While one would hope that using USB drives that have been "found" would be an anathema to any security-concious individual, there is unfortunately some reason to believe this threat is more real than we might wish to believe.

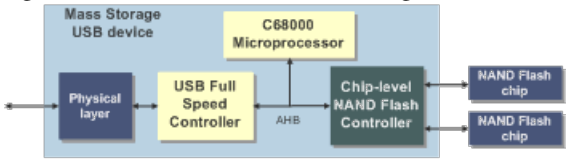
A survey of 300 IT professionals conducted by AhnLab (South Korea's largest IT security vendor) at the 2013 RSA Conference revealed that 78 per cent admitted to picking up and plugging in USB flash drives found abandoned or lying around [15]. 'Tis to weep.

No reference to "found" USB drives would be complete without mentioning the theory that the Stuxnet malware was introduced to Iran's air-gapped nuclear material processing systems by way of USB keys dropped in the parking lot (see, for example, Cherry [16]).

VIII. USB Specific Threats

In addition to the threats listed above which apply to any removable media (e.g. CD-RW), there are threats unique to USB devices.

USB storage devices (whether rotating or flash) are all built using an architecture similar to the diagram below.



As this shows, there is a microprocessor that acts as an intermediary between the USB interface and the storage itself. Normally this processor (and firmware) handle such things are wear levelling (compensating for the limited lifetime of flash storage by ensuring write operations are distributed across all blocks).

One can easily imagine the firmware designed with a hostile intent, for example:

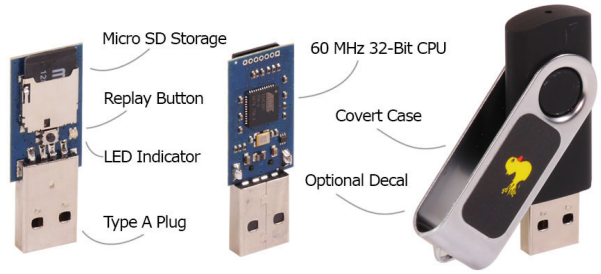
[5] Such as 1DSD3B3uS2wGZjZAwa2dqQ7M9v7Ajw2iLy. See www.flexcoin.com, archived by WebCite® at http://www.webcitation.org/6O0qzWGmj

[6] http://www.scmagazine.com/ibm-distributed-infected-usb-drives-at-conference/article/170862/

- to exploit bugs in the USB driver code by sending malformed USB protocol information during the initial device scan by the operating system;
- to hide certain files during the scan process but reveal them once the drive is plugged into the air-gapped computer;
- to insert malware into files stored on the device; or
- to appear like a normal storage drive but at some point, change into a USB keyboard device and enter commands to enable the (hidden) malware.

The USB Rubber Ducky[7] is as an example of this type of threat.

As can be seen, the unit looks like a USB flash drive one might obtain from any number of manufacturers but contains a programmable processor:



IX. Sourcing USB Storage Devices

As the above sections ought to have made clear, ensuring the provenance of USB drives used to cross the air gap is critical.

"Found" USB drives, including promotional and give-away drives ("booth bling") are to be disposed of without a second thought. Drives ought to be purchased from retail outlets, picking the model at random and carefully checking the plastic packaging to ensure it has not been tampered with (and only purchasing units that come in packaging that would be likely to show any attempts at tampering).

Proper sourcing will mitigate the following threats:

- pre-infected media (the "Stuxnet" threat)
- morphing USB devices (the "USB Rubber Ducky" threat)

X. USB Sheep Dip Appliance

As our primary defence[8] against the malware threat crossing the air gap on a USB drive we are proposing a dedicated system for scanning USB drives before they cross the air gap. We call this system a Sheep Dip Appliance (SDA).

The concept of a computer dedicated to testing removable media for malware dates back to the age of floppy disks and has become known as a *sheep dip* after the practice of removing parasites from sheep by immersing them in pesticide (see [17] [18]).

Our contribution to the art includes the following enhancements:

- using a low-cost ($< $50) ARM appliance;
- copying files (not just scanning) so that the same USB drive is not plugged into both the protected and unprotected computers; and
- processing media in both directions (inbound and outbound) and not just inbound as is normally the case.

The SDA checks the file system on the USB drive prior to attempting to mount the drive and will refuse to mount the drive if any errors are detected. By copying files to a sanitized USB drive we mitigate the threat from malformed file systems

A. The ARM Appliance

There were two objectives for the sheep dip appliance: be inexpensive so there are few impediments to its widespread deployment (we see the unit being used in non-air-gapped environments where USB drives are used to transport data within the organisation); and use a non-IA86 architecture system running Linux.

Since most self-propagating malware is designed to infect homogenous populations, using an intermediary system that is running a different processor and operating system than the donor system (in most cases) will decrease the probability the malware can infect the donor computer, then the USB, and then the sheep dip system.

There are a number of systems that meet these requirements including the Raspberry Pi, the Pogoplug, and BeagleBoard (to pick but 3).

Our proof-of-concept system was built on the Pogoplug reflashed with archlinux ARM[9] which (at least at the time this research was conducted) were widely and inexpensively ($25 - $50) available.

[7] https://hakshop.myshopify.com/collections/usb-rubber-ducky

[8] The author is Canadian and stubbornly insists on Canadian spelling.

[9] http://archlinuxarm.org/platforms/armv5/pogoplug-series-4

The specifications of this unit include:

- Processor: Marvell Kirkwood 800MHz (ARMv5te architecture)
- RAM: 128MB
- NAND: 128MB
- USB: v2

The unit also has a Gigabit Ethernet port but apart from the initial archlinux installation, the port is not used (the sheep dip appliance is itself air-gapped).

To provide a user interface to enable selection of the desired operation (see the next section) we have added an LCD display and keypad from Matrix Orbital[10]. A simple C program has been written to control the display and run the scripts to implement the SDA's functionality[11].

As an aside, it is somewhat ironic that the ARM "server" that does all the work can be obtained for $25 while the display and keypad unit costs $100. There are less expensive display/keypad units available but they typically come without a protective case.

To complete our SDA we use a small four-port USB hub to connect the single USB port on the Pogoplug to the display/keypad and (up to two) USB drives.

B. Sheep Dip Functionality

The current implementation supports the following main functions:

1. Sanitise USB Drive
   The SDA will write zeroes to every block on the device (which can take several minutes) and then format the empty drive.
2. Copy USB Drive
   The SDA expects two USB drives to be inserted, one that is empty (and has been sanitized by the SDA), and the other containing files to be scanned and copied. The SDA will validate the file system structure of the source USB drive (i.e. *fsck*), copy the contents of the files to the sanitized USB drive, and then scan them using ClamAV[12].

A second level admin menu provides the following options:

1. Update AV Files
   The SDA expects to find a USB drive containing updated signature files for ClamAV and will update the antivirus software.
2. Download SDA Log
   The SDA expects to find an empty sanitized USB drive and will copy the log files from the SDA to the drive.
3. Update SDA
   The SDA expects to find a USB file containing a signed script file that will be executed to update the SDA's software.

All operations are logged with the serial number of the drive(s) involved and the files copied.

C. Sheep Dip Procedure

In the following procedure we shall copy files from an unprotected computer across the air gap to a protected computer. For transfer of files in the reverse direction (from the protected to the unprotected), just reverse the process. Note that this is a significant addition to the process suggested by most documents on air-gapped networks that assume that it safe to take a USB drive directly from the protected computer to the unprotected (i.e. only use the sheep dip in one direction).

Step 1. Prepare two USB drives
To securely copy files to a protected computer one prepares by using the SDA to sanitize two brand new USB flash drives (see the section on sourcing USB devices). One is then marked "unprotected" and the other "protected" (the use of differently coloured USB drives will help).

Step 2. Put files on the unprotected USB drive
The files to be transferred across the air gap are put on the USB drive (if the files originate from the protected computer, they are put on the USB drive marked "protected", otherwise vice versa).

Step 3. Use the SDA to copy and scan files
The two USB drives (one with the files to be transferred across the gap and the other an empty sanitized drive) are put into the SDA and the contents copied to the empty USB drive and then scanned using anti-virus software. Any files that the AV software considers malware are deleted and an error is displayed and logged.

Step 4. Copy files to the protected system
Finally the "protected" USB drive is inserted into the protected system. Et voilà.

XI. The Reverse Path

The counterpart to the risk of malware crossing the air gap from the "outside world" to the protected (air-gapped) systems is malware exploiting removable media to exfiltrate information from the air-gapped system to the outside world.

One might ask why this vulnerability need be considered if we have been rigorous in protecting our air-gapped systems from becoming infected with malware.

Three words. Defence in depth.

While the USB sheep dip, when used properly, will scan all files that are transported across the air gap to the protected systems, AV scanning is not a perfect science. We must also consider the possibility that the protected systems were infected by personnel who bypassed the sheep dip (either out of ignorance, a sense of expediency, or malice).

[10] http://www.matrixorbital.com/External-LCDs-USB-Character-ELK/c53_52/index.html

[11] Software available at http://github.com/pajari/usbsheepdip

[12] http://www.clamav.net/

Various means of mitigating this vulnerability have been proposed. Schneier[13] has recommended using the smallest capacity USB drive possible or to use optical media in order to to limit the amount of "unexpected" (exfiltrated) data that might "piggyback" on the USB as it returns across the air gap.

We propose an alternative approach. Our recommended procedure is to require USB devices to pass the sheep dip in the reverse direction (not only to the protected system but from the protected system).

This will prevent hidden data (i.e. data stored in unallocated areas) from being transferred. Also the SDA displays the number of files and the total amount of data copied (which the user ought to compare to the expected values).

XII. Conclusion

True air-gapped computers may be a myth but disconnected computers that use a carefully crafted procedure that incorporates a sheep dip appliance for transfers across the gap *(*in both directions) can provide considerably more security than depending on firewalls and unified threat management devices.

Appendix A
Software

The complete software (source and scripts) for the Sheep Dip Appliance will be made available shortly on GitHub[14] along with the most current revision of this paper (and any related documents).

Appendix B
Recommended Policies

The proposed sheep dip, as with any risk mitigation strategy, can only realise its full potential by operating in the context of documented, enforced, and audited information security management policies and procedures.

Some recommended policies might address:

Physical Security
: Any mechanisms to protect systems by air-gapping is rendered moot if your adversaries have physical access to the server.

No Lone Zone
: Rules (possibly enforced using physical security controls) that prohibit a single person from working in the area of the protected computer without supervision[15].

USB Drive Provenance
: Ensuring that only USB drives obtained from reliable sources in a manner to preclude undetected tampering are used.

Use of the USB Sheep Dip
: Ensuring that all staff that are responsible for moving data across the air gap understand the risks, the importance of using the sheep dip appliance, and how to properly use it (e.g. drives must be "dipped" in *both* directions; file copying statistics must be checked against expectations).

Decommissioning of Air Gap USB Storage Devices
: Ensuring that USB drives that have reached their end of life are properly disposed of (see below).

Appendix C
Decommissioning USB Flash Drives

Although the sheep dip appliance described in this paper sanitizes USB storage devices by writing zeroes to all addressable blocks, this ought not to be considered sufficient to destroy all data on the drive owing to the unique characteristics of flash storage (see Wei et al [19]).

Once a USB flash storage device has been used to store sensitive information, the only way to safely decommission the device is to physically destroy it (e.g. by incineration or the judicious use of a large sledgehammer[16]).

Appendix D
Further Work

There are a number of ways in which this work could be extended. The current design does nothing to mitigate insider threats. A more complete design would enforce policies to ensure that

- USB drives designated for use on the "unprotected" side of the air gap are never mounted on protected computers
- files are only copied onto the protected computers if they have been scanned by the SDA

For example, one could set up a second Pogoplug (or similar) unit networked with the protected computer(s) to function as a file server. The protected computers would be configured not to mount any USB removable drives directly (see NSA [20]) and the Pogoplug would be configured to only automount USB devices that were recognised "protected" USB devices and that contained files that had been scanned by the SDA (as evidenced by being signed by the SDA). Once varified, the drive would be made available to the protected computer using a network file sharing protocol such as SMB or NFS.

In environments in which only certain types of files typically cross the air gap, one could extend the SDA software to only copy files of the expected/permitted types (recognised either by file name extension and/or file content examination in a manner similar to the UNIX file command [21].

If the type(s) of files being moved to the protected computer is different from those being moved from the protected computer one could extend the SDA to use a different set of file type filters depending on the direction of the transfer.

Acknowledgments

The author would like to thank: Eric Byres (Tofino Security), Devin Kinch, and Bruce Matsugu for reviewing an early copy of this paper. Particular thanks to Dragos Ruiu, and the organisers of the CanSecWest 2014 conference in Vancouver

[13] https://www.schneier.com/blog/archives/2013/10/air_gaps.html

[14] https://github.com/pajari/usbsheepdip

[15] http://en.wikipedia.org/wiki/Two-man_rule

[16] Remember to wear your safety glasses.

for the opportunity to present a talk based on this research. This work has also benefitted from discussions with several of the people at Bex.io. Words are insufficient to express my appreciation to my wife for her understanding, patience, and love (not to mention her unparalleled editing skills).

REFERENCES

[1] Wikipedia, "Computer security." [Online]. Available: http://en.wikipedia.org/wiki/Computer_security

[2] Harvard University (Office of the Vice Provost for Research), "Harvard research data security policy (HRDSP) level 5 requirements," 2010, [accessed 15-October-2013]. [Online]. Available: http://security.harvard.edu/filebrowser/HRDSPLevel5requirements10110.pdf

[3] The VirtEx Difference. [Online]. Available: https://www.cavirtex.com/why_virtex

[4] Wikiquote, "Gene Spafford," 2014, [accessed 5-March-2014]. [Online]. Available: http://en.wikiquote.org/wiki/Gene_Spafford

[5] E. Byres, "#1 ICS and SCADA Security Myth: Protection by Air Gap." [Online]. Available: http://www.belden.com/blog/industrialsecurity/1-ICS-and-SCADA-Security-Myth-Protection-by-Air-Gap.cfm

[6] B. Schneier, "Want to evade NSA spying? don't connect to the internet," *Wired*, oct 2013. [Online]. Available: http://www.wired.com/opinion/2013/10/149481/

[7] D. Gilbert, "International space station infected with USB stick malware carried on board by russian astronauts." [Online]. Available: http://www.ibtimes.co.uk/articles/521246/20131111/international-space-station-infected-malware-russian-astronaut.htm

[8] Wikipedia, "Tempest (codename)," 2014. [Online]. Available: http://en.wikipedia.org/wiki/Tempest_(codename)

[9] M. Hanspach and M. Goetz, "On covert acoustical mesh networks in air," *Journal of Communications*, vol. 8, no. 11, nov 2013. [Online]. Available: http://www.jocm.us/index.php?m=content&c=index&a=show&catid=124&id=600

[10] D. Goodin, "Meet "badBIOS", the mysterious Mac and PC malware that jumps airgaps." [Online]. Available: http://arstechnica.com/security/2013/10/meet-badbios-the-mysterious-mac-and-pc-malware-that-jumps-airgaps/

[11] D. E. Sanger and T. Shankerjan, "N.S.A. devises radio pathway into computers," *New York Times*, p. A1, Jan 14 2014.

[12] Wikipedia, "Faraday cage." [Online]. Available: http://en.wikipedia.org/wiki/Faraday_cage

[13] Microsoft, "MS13-027: addressing an issue in the USB driver requiring physical access," [Online]. Available: http://blogs.technet.com/b/srd/archive/2013/03/12/ms13-027-addressing-an-issue-in-the-usb-driver-requiring-physical-access.aspx

[14] J. Larimer, "Beyond AutoRun: Exploiting vulnerabilities with removable storage." [Online]. Available: https://media.blackhat.com/bh-dc-11/Larimer/BlackHat_DC_2011_Larimer_Vulnerabiliters_w-removeable_storage-wp.pdf

[15] MarketWire Press Release, "AhnLab Survey: 78% of IT Professionals Admit Picking Up and Plugging In Abandoned USB Drives," March 2013, http://www.marketwired.com/press-release/ahnlab-survey-78-it-professionals-admit-picking-up-plugging-in-abandoned-usb-drives-1769319.htm.

[16] S. Cherry, "How Stuxnet is rewriting the cyberterrorism playbook," *IEEE Spectrum (podcast)*, oct 2010. [Online]. Available: http://spectrum.ieee.org/podcast/telecom/security/how-stuxnet-is-rewriting-the-cyberterrorism-playbook

[17] Wikipedia, "Sheep dip (computing)," http://en.wikipedia.org/wiki/Sheep_dip_(computing).

[18] PSN Infrastructure Security & Cyber Defence Team, UK Cabinet Office, "Common standard for malware protection," 2012. [Online]. Available: https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/81648/Malware_Protection_Std_v1-0_0.pdf

[19] *Reliably Erasing Data From Flash-Based Solid State Drives*, feb 2011. [Online]. Available: https://www.usenix.org/conference/fast11/reliably-erasing-data-flash-based-solid-state-drives

[20] National Security Agency, "Disabling USB Storage Drives." [Online]. Available: http://www.nsa.gov/ia/_files/factsheets/i731-002r-2007.pdf

[21] Wikipedia, "File (command)." [Online]. Available: http://en.wikipedia.org/wiki/File_(command)

George E. Pajari George E. Pajari is principal of HCIS Health Care Information Security, an information security consultancy and managed security service provider specialising in the health care sector. Previously he set up the first Network Operations Centre (NOC) at GLENTEL Inc., a company that builds and maintains radio communications networks for industry and government, including police and fire departments.

He holds the $(ISC)^2$ CISSP and ISACA CISM certifications in information security. He obtained his undergraduate degree in Computing and Information Science from Queen's University at Kingston and has pursued graduate studies at the University of British Columbia. He has taught at UBC and the University of Calgary.

His book on *Writing UNIX Device Drivers* was published by Addison-Wesley.

Email: George.Pajari (you-know-what-goes-here) HCIS.ca; Twitter: @OrangeHazMat; Blog: http://orangehazmat.wordpress.org/