IoTMU: Smart Protection of Smart Homes

1st Youngjun Park
Electrical and Computer Engineering
Air Force Institute of Technology
Wright-Patterson AFB, USA
youngjun.park@afit.edu

2nd Richard Dill
Strategic Studies
Air Force Institute of Technology
Wright-Patterson AFB, USA
richard.dill@afit.edu

3rd Barry E. Mullins
Electrical and Computer Engineering
Air Force Institute of Technology
Wright-Patterson AFB, USA
barry.mullins@afit.edu

***Abstract*—In the changing landscape where increasing number of organizations are deploying smart devices in their networks, one of the greatest challenges they ~~must~~ face is security. While the use of Internet of Things (IoT) enabled new capabilities such as ease of access, remote control, and interoperability, it also enabled a multitude of new attack vectors for adversaries to exploit. ~~The~~ IoT devices are equipped with limited hardware capacity to effectively carry out the additional computation required for security such as data encryption. As a result, gaining access to the data associated with the IoT devices becomes almost trivial as long as the third-party entity has physical access to the device or the network ~~with~~ which the devices are connected. In addition, although these vulnerable devices are increasingly being integrated into the operations of various organizations, such as the U.S. Department of Defense, the necessary policies to safeguard these inherently vulnerable devices are lacking behind. Unfortunately, the production of the innately vulnerable devices cannot be effectively regulated without a governing policy and the immediate burden of securing the devices often fall on the end users themselves. To help mitigate the vulnerabilities stemming from the hardware limitations of IoT devices, ~~here~~ we present a defensive model to obfuscate the sensitive data sent over Wi-Fi. As a proof of concept, we show that the video stream created by one of the most popular IoT cameras being sold on Amazon can be recreated via passive sniffing. Then we propose a central management agent which acts as the proxy for the vulnerable IoT devices to both obfuscate the network traffic by mimicking real devices, and to serve as an encryption agent for the devices with limited computational capacity. The model requires minimum set up for the users, and is compatible with any device that is configurable over Wi-Fi.**

***Index Terms*—Internet of Things (IoT), data security, network traffic obfuscation, reverse engineering**

I. Introduction

The term Internet of Things (IoT) represent wave of embedded technologies with the added functionality of connectivity. Since its inception, the model has infiltrated numerous public sectors ~~to make up~~ Industrial Control Systems (ICS), cities, healthcare, and government [1]; according to the International Data Corporation, the IoT industry is projected to reach 1.2 trillion dollars by the year 2021 [2]. However, the rapid growth of the industry was rivaled by the increasing number of vulnerabilities that were discovered in the devices. As the devices continued to be deployed in critical infrastructures and national institutions, investigating secure policies for the use of IoT became one of the priorities for organizations such as the U.S. Department of Defense [3].

~~Pre-existing~~ embedded systems found in automotives, Supervisory Control and Data Acquisition (SCADA) systems, among others were originally designed to function as closed systems. As a result of connecting those systems to the outside world, the devices were subject to the numerous vulnerabilities they werent designed to be protected against. The dangers of these design deficiencies are highlighted in a 2015 study on the infotainment system found in modern vehicles discovered a vulnerability that allowed an adversary to gain remote control of the vehicle [4]. Security experts have recognized the security flaws of the technology and investigated potential attack surfaces of IoT devices to help the manufacturers and the users to mitigate them, including the effort by the Open Web Application Security Project (OWASP) [5]. However, the physical limitations of the devices often become the bottleneck for meaningful security measures such as encryption which requires large computational power. Security is therefore left in large part to the end users ~~to practice~~.

Currently one of the most common mode of communication for IoT devices is Wi-Fi [1]. While the security of Wi-Fi has improved after transitioning from the WEP standard to WPA2, there still exists vulnerabilities that allow an adversary to gain access to the network through publicly available password cracking tools such as Aircrack-ng [6] and Cain [7]. After gaining access to the network, the attacker can passively sniff the network traffic from afar. One of the vulnerabilities of an IoT device is its data being sent in the clear, allowing an attacker to eavesdrop on the connection. In this study we reverse engineer one of the most popular wireless cameras on Amazon to illustrate its eavesdrop vulnerability. Numerous studies so far have demonstrated the vulnerabilities that exist in network cameras including those of [8-10]. In particular, the 2009 DEFCON 17 talk showcases a series of attacks that can be taken against IoT cameras via ARP cache poisoning, a common technique to eavesdrop on network traffic between hosts. This study highlights the pervasiveness of eavesdrop vulnerability via passive network scan 10 years after the DEFCON talk.

Over the years, researchers have sought out ways to mitigate the inherent security threats present in IoT networks. These approaches include local area network (LAN) management schemes via software defined networks (SDN) [11, 12], deployment of encryption gateway [13], and obfuscating network traffic by sending crafted traffic [14]. While the SDN approach

prevents a compromised device or a malicious host from further attacks, it does not prevent a bystander from passively eavesdropping on the network traffic. Using encryption gateways prevents an adversary from eavesdropping on sensitive data such as those of IoT cameras, however their proposed methods require physical connection of the vulnerable devices to the gateway via a serial connection which is not a practical setup in modern smart networks. The authors of [14] were able to defeat device fingerprinting and information leakage in a smart home by spoofing Wi-Fi traffic. This study presents a model which combines the encryption gateway and network traffic spoofing to enhance network confidentiality in an IoT network.

This studys contributions are as follows:

- We demonstrate an eavesdrop vulnerability in a popular IoT camera sold on Amazon
- We present an automated tool to extract MJPEG stream from network traffic
- We introduce IoTMU: a data confidentiality model in IoT network that performs network traffic obfuscation and application level encryption

II. Threat Model

While an IoT camera has its use in many facets of industry, this study focuses its use in a smart home network. A typical smart home network consists of a router that serves as the access point (AP) to directly connect its constituent devices to the internet via Wi-Fi. Other modes of wireless communications for IoT such as Zigby [15], and Bluetooth [16] exist, but they are out of scope of this study. The threat model (Figure 1) of this study consists of the following:

- A smart home network set up by a user which consists of a central AP to connect different IoT devices via Wi-Fi secured with WPA2
- A user accessing the video feed from an IoT camera from a remote location via an application provided by the vendor
- An adversary in proximity to the smart home who has gained access to the network by cracking the WPA2 preshared key and passively sniffing the network

Our assumption of an adversary gaining access to the network is substantiated by numerous research that investigate password cracking on Wi-Fi networks including [17, 18]. After gaining access to the network the adversary is able to passively sniff the network traffic and analyze them without detection through tools such as Wireshark [19]. Because many IoT devices send data to the network without encryption [5] the adversary is able to gain access to sensitive information without gaining access to the devices themselves. As highlited in [20], there are numerous attacks that an adversary can perform after gaining access to a target network. However this study focuses on compromise of data confidentiality as a result of eavesdropping in an unprotected network.

The primary goal of IoTMU is protection of sensitive data as a result of insecure transportation of data. IoTMU is a

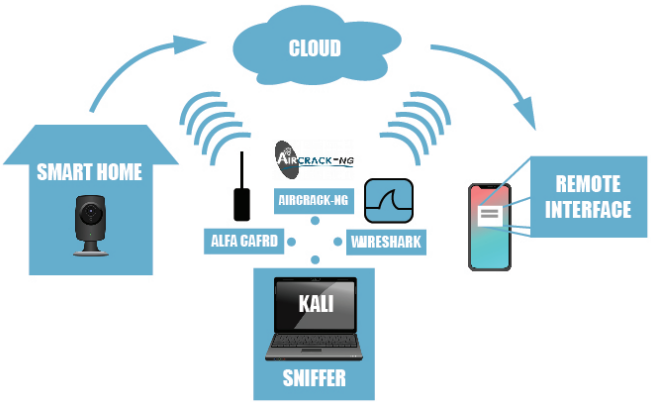

Fig. 1. Overview of adversarial model

security agent that sits between the devices and the router that perform encryption for the computationally constrained end devices. It is paired with a decryption agent on the other end of the communication that sits on the device the user uses to interact with the IoT devices. The IoTMU also performs periodic spoofing to deter device fingerprinting and obfuscate application-level data.

III. Investigating the vulnerabilities of an IoT camera

As a proof of concept we investigate the vulnerability of the network protocol for Wansview Wireless 1080P Security Camera model Q3 being sold on Amazon [21]. As of May 15, 2019 it holds the Amazons Choice label on the website for the keywords wifi baby monitor with more than 2,500 customer reviews of average 4.1 out of 5-star rating scale.

A. Experimental Setup

For analysis we use a network interface card set to monitor mode to capture wireless traffic and wireshark to decode the captured data. The IoT camera is connected to the router acting as an AP via Wi-Fi and communicates with the vendor application running on an Android device over 4G network provided by the cellular provider. In proximity to the smart home is an adversary on a laptop running Kali as its operating system. An Alfa card is connected to the laptop and configured to monitor mode to capture the network traffic in the smart home. For the purpose of this study we assume the adversary has performed the necessary reconnaissance and analyses to determine the MAC and IP addresses of the camera and the router, and was able to gain access to the WPA2 pre-shared key via social engineering and/or password cracking. A comprehensive list of devices, tools and configuration are summarized in Table 1 (TODO). There are methods publicly available to gain root access to the camera with its admin credentials [22], which has been verified by the author. However, this study focuses on passive network sniffing which does not require direct interaction with the device.

B. Methodology

In the sniffing setup, the Alfa card connected to the laptop was first set to monitored mode via airmon-ng. Then airodump-ng was used to identify the AP of interest. While using airodump-ng the second time to record the network traffic associated with the target AP, aireplay-ng was used to send spoofed deauthentication messages to the IoT camera to capture the extensible authentication protocol over LAN (EAPOL) 4-way handshake between the AP and the camera. The captured EAPOL key is used to decrypt the Wi-Fi traffic between the AP and the router. If stealth is a key consideration, instead of sending deauthentication messages which could trip an alarm in a network anomaly detection system, attacker could have waited until the camera is manually turned on and off by the user via social engineering. While the attacker is gathering network data, the user in a remote location accessed the camera feed through the mobile application provided by Wansview on an Android phone. After a period of time the attacker stopped the sniffer and viewed the recorded data for analysis on Wireshark. In order to view the encrypted Wi-Fi data on wireshark, the preshared key previously acquired by the attacker was inputted under the IEEE 802.11decryption key setting.

C. Analysis

The camera first performed a Domain Name System (DNS) lookup of its cloud server followed by a series of network discovery protocols including Simple Service Discovery Protocol (SSDP) and Medium Independent Network Transport (MiNT) protocol . Its primary mode of communication was User Datagram Protocol (UDP) which is often used for transportation of time-sensitive data like video streams [23].

Determining the initial start time of video stream was clear, highlighted by the jump in the length of payload from mostly sub-100 range to 1,032. At first glance, Wireshark was unable to determine the type of data being transmitted. However, exporting one of the 1,032 payload and analyzing its entropy showed a steep downward slope suggesting that the payload is not encrypted (Figure 2). Motivated by our finding, a deeper inspection of the first packet of the stream revealed the file signature of a JPEG image, or JFIF in ascii characters. Further investigation of subsequent packets showed that the video was transferred as an MJPEG stream suggesting that the initial JPEG image corresponds to the thumbnail image shown in the Android application.

Once the video stream began, on top of the stream data, the UDP payloads also contained various control information consisting of a required 8-byte block and an optional block that varied from 0 to 40 bytes. Figure 3 illustrates the breakdown of the UDP payload. The first byte of the payload was a fixed value of 0xf1, which was followed by either 0xd0, 0xd1, 0xe0, 0xe1. 0xd0 was used in payloads with stream information, whereas 0xd1 seemed to serve as acknowledgement packets analogous to the Transmission Control Protocol counterpart (TCP). 0xe0, and 0xe1 were sent out by both the server and the client, always followed by a two-byte zero padding, likely

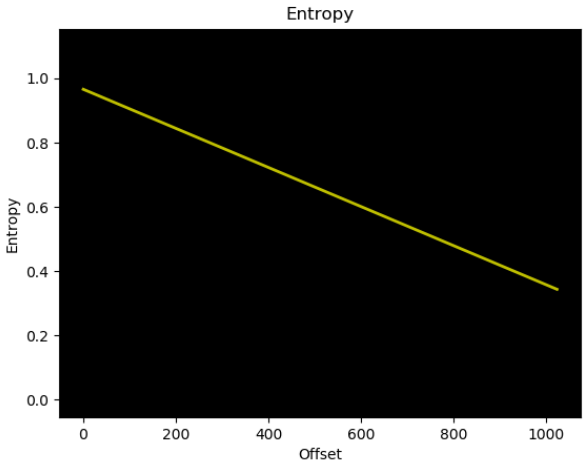

Fig. 2. Entropy of a sample payload during video stream

representing a type of keep alive signal to keep the stream open or a watchdog function. The next two varying bytes represented the length of the payload following the two bytes (i.e. length of payload in bytes - 4 bytes). They were followed by a 0xd100 for the any data part of the JPEG image, 0xd101 for data part of the MJPEG stream and 0xd102 for all others. The final two bytes in the required 8-byte header represented the sequence number of the data to follow, which made up for the lack of ordering information present in TCP.

There were many variations of the optional overhead following the required 8-byte header depending on the type of packet. However for the purpose of extracting the video feed we were only required to determine that the optional header for the packet containing the JPEG header was 8 bytes, and the optional header for the packet containing the MJPEG header was 32 bytes. There were, however, optional 32-byte headers for MJPEG packets that were not the first in the series of packets. These headers could be distinguished by the 0x55aa sequence following the 8-byte header and pertinent data could be correctly extracted by filtering for the specific sequence. Finally, any stream data following the initial headers did not contain any optional headers. The aforementioned header information were used to build an automated MJPEG extraction tool (Appendix 1) to recreate the video stream (Figure 4) without having physical access to the camera or its credentials.

Due to the inconsistent nature of Wi-Fi traffic, the reconstructed video feed was not a perfect replica. However, most of the feed was recognizable as stream from the mobile application. As previously identified [5] the eavesdrop vulnerability found in this study showcases its pervasiveness in IoT devices. This security flaw can easily be taken advantage of by a malicious insider or a determined adversary; it warrants further research into various mitigation techniques in the immediate future.

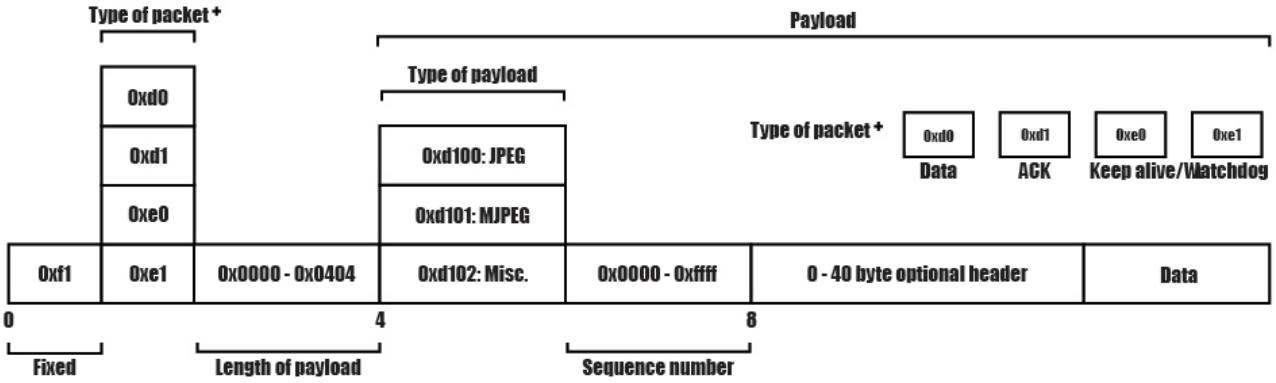

Fig. 3. Breakdown of a UDP packet payload during camera stream

Fig. 4. Snapshot of the reconstructed video stream

IV. IoTMU Design

To mitigate the eavesdrop vulnerability created by unencrypted application-level traffic, we propose IoTMU, a central IoT gateway. Its setup in a typical smart home is depicted in Figure 5. In a typical end-to-end communication between hosts, there are two sides of traffic to protect. However, in a smart home setting the easier of the two end hosts is the smart home end of the communication because the devices are often stationary and therefore static in its network.

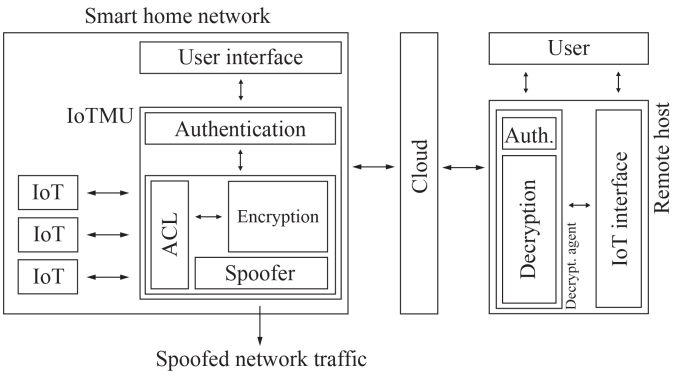

Fig. 5. Snapshot of the reconstructed video stream

To protect this vulnerable end of the communication IoTMU will serve as the proxy to all IoT devices present in the network to encrypt its application level traffic before forwarding it to the router. IoTMU will be paired with a decryption agent living on the other host, often with enough computational capacity to perform encryption and decryption (i.e. computer, smartphone) by itself. IoTMU will consist of the following components: (1) Authentication, (2) Access control list (3) Encryption, and finally (4) Spoofing.

The encryption agents will be accessed and configured through an authentication mechanism via username and password set by the user. In addition, to ensure only the intended devices are communicating with the hub, IoTMU will store an access control list based on the network signatures of the IoT devices such as MAC address, IP address, port number, etc. This does not in fact prevent an adversary from spoofing one of the IoT devices to communicate with the gateway. But since the intention of the gateway is data confidentiality, it will not affect its functionality.

Similar to the approach taken by [13], the encryption can be performed through a public key encryption (PKI). This requires the exchange of keys between IoTMU in the smart home network and each of the encryption agents residing in the other end host. The encryption agents for the end hosts can take the form of a smart phone application that runs in the background or an executable on a computer. Initially, each participating hosts will exchange their public keys signed by a common entity via symmetric key encryption. All subsequent traffic will be encrypted and decrypted using the private key and the public key of the end hosts. The use of PKI will prevent a bystander from intercepting a common key to decrypt the messages. Although taking an approach like SSL using PKI to exchange session keys and using symmetric keys for later exchanges may lesson the computational demand, its vulnerability to man-in-the-middle attacks [24] defeats the purpose of IoTMU.

Finally, the IoTMU will periodically send out spoofed Wi-Fi traffic to mimic certain device types. Spoofing network traffic substantiated by the research in [14], will help fortify the

network against fingerprinting and information leakage in a smart home. It will also protect the unencrypted data being exchanged between IoTMU and the IoT devices. Creating of spoofed packets can be variations on the following criteria:

- Length of the payload in the network
- Frequency and timing of the packets sent
- Spoofing unused IP address in the network
- Spoofing a plausible MAC addresses of certain vendors to mimic device types

Using the aforementioned criteria, a central IoT gateway design will help secure a smart home network without changing any existing protocols or relying on the vendors for security. However, the latency and the packet overhead imposed by encryption and the effect of periodic spoofing on network congestion is left for future investigation.

V. Conclusion

This paper discussed the vulnerabilities of IoT devices as it exists in a smart home network. It demonstrated an existing vulnerability in a popular IoT camera by reverse engineering its communication protocol and by creating an automated tool to extract the MJPEG stream created by the camera. The proposed design of IoTMU mitigates this vulnerability for the IoT devices in smart homes through encryption and spoofing. As it does not rely on any existing protocols, IoTMU can be implemented for the varying protocols utilized by the IoT devices. Implementation of the model and the analysis of its performance is left a future work.

References

Unless there are six authors or more give all authors' names; do not use "et al.". Papers that have not been published, even if they have been submitted for publication, should be cited as "unpublished" [4]. Papers that have been accepted for publication should be cited as "in press" [2]. Capitalize only the first word in a paper title, except for proper nouns and element symbols.

References

[1] P. P. Ray, "A survey on Internet of Things architectures," J. King Saud Univ. - Comput. Inf. Sci., vol. 30, no. 3, pp. 291319, 2018.

[2] M. Torchia and M. Shirer, "IDC Forecasts Worldwide Technology Spending on the Internet of Things to Reach $1.2 Trillion in 2022," International Data Corporation, 2018. [Online]. Available: https://www.idc.com/getdoc.jsp?containerId=prUS43994118.

[3] United States Government Accountability Office, "INTERNET OF THINGS Enhanced Assessments and Guidance Are Needed to Address Security Risks in DOD," 2017.

[4] C. Valasek and C. Miller, "Remote Exploitation of an Unaltered Passenger Vehicle," Black Hat USA, vol. 2015, pp. 191, 2015.

[5] Open Web Application Security Project, "IoT Attack Surface Areas," 2015. [Online]. Available: https://www.owasp.org/index.php/IoT_Attack_Surface_Areas.

[6] "Aircrack-ng," 2018. [Online]. Available: https://www.aircrack-ng.org.

[7] "Cain and Abel," 2014 [Online]. Available: https://www.oxid.it/cain.html

[8] C. Heffner, "Exploiting Surveillance Cameras Like a Hollywood Hacker," no. February, p. 30, 2013.

[9] J. Ostrom and A. Sambamoorthy, "DEFCON 17: Advancing Video Application Attacks with Video Interception, Recording, and Replay," 2009. [Online]. Available: https://www.youtube.com/watch?v=QcsQ6UzMJiU.

[10] N. Kalbo, "I Got My EyeOn You - Security Vulnerabilities in D-Links Baby Monitor," 2018. [Online]. Available: https://dojo.bullguard.com/dojo-by-bullguard/blog/i-got-my-eyeon-you-security-vulnerabilities-in-baby-monitor/.

[11] M. Miettinen et al., "IoT Sentinel Demo: Automated Device-Type Identification for Security Enforcement in IoT," Proc. - Int. Conf. Distrib. Comput. Syst., pp. 25112514, 2017.

[12] D. Soteris et al., "SDN-driven protection of smart home WiFi devices from malicious mobile apps," Proc. 10th ACM Conf. Secur. Priv. Wirel. Mob. Networks, pp. 122133, 2017.

[13] C. Doukas, I. Maglogiannis, V. Koufi, F. Malamateniou, and G. Vassilacopoulos, "Enabling data protection through PKI encryption in IoT m-Health devices," IEEE 12th Int. Conf. Bioinforma. Bioeng. BIBE 2012, no. November, pp. 2529, 2012.

[14] S. M. Beyer, B. E. Mullins, S. R. Graham, and J. M. Bindewald, "Pattern-of-Life Modeling in Smart Homes," IEEE Internet Things J., 2018.

[15] Zigbee Alliance, "What is Zigbee?," 2018. [Online]. Available: https://www.zigbee.org/what-is-zigbee/

[16] Bluetooth SIG, "Protocol Specifications," 2019. [Online]. Available: https://www.bluetooth.com/specifications/protocol-specifications/

[17] O. Nakhila and C. Zou, "Parallel Active Dictionary Attack on IEEE 802.11 Enterprise Networks," in MILCOM 2016-2016 IEEE Military Communications Conference, 2016, pp. 265270.

[18] A. Abdelrahman, H. Khaled, E. Shaaban, and W. S. Elkilani, "WPA-WPA2 PSK Cracking Implementation on Parallel Platforms," in 2018 13th International Conference on Computer Engineering and Systems (ICCES), 2018.

[19] "Wireshark," 2019. [Online]. Available: https://www.wireshark.org/

[20] M. Nawir, A. Amir, N. Yaakob, and O. B. Lynn, "Internet of Things (IoT): Taxonomy of security attacks," in 2016 3rd International Conference on Electronic Design, 2016, pp. 321326.

[21] "Wansview Wireless 1080P Security Camera, WiFi Home Surveillance IP Camra for Baby/Elder/Pet/Nanny Monitor, Pan/Tilt, Two-Way Audio & Night Vision Q3-S," Amazon, 2019. [Online]. Available: https://www.amazon.com/Wansview-Wireless-Security-Surveillance-Monitor/dp/B075K89NTR/ref=sr_1_4?crid=3MPNIXJO00PA5&keywords=wifi+baby+monitor&qid=1558389992&s=gateway&sprefix=wifi+baby+mo%2Caps%2C159&sr=8-4

[22] J. Wedell, "Rooting a cheap IP camera (Wansview K2)," Rublin Wedells, 2017. [Online]. Available: https://jonwedell.com/rooting-a-cheap-ip-camera/

[23] J. F. Kurose and K. W. Ross, Computer networking: a top-down approach, 7th ed. New Jersey: Pearson, 2017.

[24] E. Skoudis and T. Liston, Counter hack reloaded: a step-by-step guide to computer attacks and effective defenses, Prentice Hall Press, 2005.