Journal in Computer Virology and Hacking Techniques manuscript No.
(will be inserted by the editor)

Evolution and Characterization of Point-of-Sale RAM Scraping Malware*

Ricardo J. Rodríguez

the date of receipt and acceptance should be inserted later

Abstract Credit and debit cards are becoming the primary payment method for purchases. These payments are normally performed in merchant's in-store systems as known as Point-of-Sale (POS) systems. Since these systems handle payment card data while processing the customer transactions, they are becoming a primary target for cybercriminals. These data, when remain at memory, are scraped and exfiltrated by specially crafted malicious software named POS RAM scraping malware. In recent years, large data breaches occurred in well-known US retail companies were caused by this kind of malware. In this paper, we study the features of these malware based on their behavior on different stages: infection and persistence, process and data of interest search, and exfiltration. Then, we classify samples of 22 known POS RAM scraping malware families from 2009 to 2015 according to these features. Our findings show these malware are still immature and use well-defined behavioral patterns for data acquirement and exfiltration, which may make their malicious activity easily detectable by process and network monitoring tools.

Keywords malware, POS RAM scraping, evolution, taxonomy, software security

* The final publication is available at Springer via DOI: 10.1007/s11416-016-0280-4.

Ricardo J. Rodríguez
Dpto. de Informática e Ingeniería de Sistemas
Universidad de Zaragoza, María de Luna 1, 50018 Zaragoza, Spain
Tel.: +34 876-55-5531
Fax: +34 976-76-1914
E-mail: rjrodriguez@unizar.es

1 Introduction

Financial services are considered a critical infrastructure sector since provides essential services to our society [12]. This sector faces a wide range of potential risks that may disrupt its normal operation. Unlike other critical infrastructures that may suffer outages by unintended events, financial services are mainly targeted by intended attacks. The number and sophistication of cyberattacks that target to this sector demonstrate an increasing interest of cybercriminals [19, 58].

In fact, credit (and debit) card data are extremely sought-after items in the underground market. As reported by Symantec, the cost of US single credit card data ranges between \$1.50 to \$5 (fortunately discounts may apply when buying in bulk), while EU card data cost \$5 to \$8 [52]. This difference is mainly based on the wide spread availability of stolen US cards. Note that we just refer to minimum card data needed to complete a payment (that is, credit card number, expiry date, and the name of the cardholder). When card details incorporate extra information (known as *fullz card* in the underground market), such as date of birth, security password, or other data that facilitate identity theft, sell prices may increase up to \$20 (for US cards). Of course, these prices also increase depending on the card type (i.e., gold, platinum, or business).

Personal card details are usually retrieved from point-of-sale (POS) devices, i.e., in-store systems used for customers to pay merchants for goods or services, among other transactions [3]. A recent summary of publicly known cyberattacks occurred on US companies during 2014 reported that 36% of them were related to stolen credit card customer information, mostly occurred at retailers or restaurants [58].

Malicious software (*malware*) are specially crafted software that aim at gaining access to computer systems with malicious intentions. Connected POS systems, as any computer system, can be targeted by malware specially designed to seek credit card data that may reside at different location within the system. These systems are becoming a recent target of cybercriminals, as numerous security vendors reported [52, 53, 54]. These attacks span from skimming terminals to network sniffers [45, 50]. Hence, data leakages as result of attacks become a common method to obtain personal card details [56].

To date, one of the largest known POS breach occurred at The TXJ Companies, Inc., at 2008 [20]: a weak encryption scheme used in the wireless network of some stores was successfully exploited to gain access to the internal network. From there, hackers obtained access to company servers that stored customer card details. It is estimated more than 40 million of credit-card records were actually stolen. In 2010, Albert Gonzalez was found guilty and sentenced to 20 years in prison for committing these felonies [63].

Network sniffers were used in that case to analyze the network communication and find out the vulnerability. In the past, network sniffers were the major threat of POS systems. However, nowadays they were relegated by RAM scraping malware [8]. These crafted malware aim at collecting the processes on exe-

cution within a system, and then scanning their allocated memory to look for patterns that match personal card details. This kind of malware have evolved so much that security vendors even offer ad-hoc software solutions [26, 52].

In particular, biggest data leakages occurred at 2013 and 2014 were caused by POS RAM scraping malware [52]. `BlackPOS` malware was behind data breach occurred at Target in 2013, which exposed 40 million of customer's credit and debit card information in only three weeks. Few months later (in Sept 2014), U.S. retailer Home Depot announced 56 million of payment card data may have been compromised in a five-month attack caused by a variant of `BlackPOS`, known as `FrameworkPOS`.

In this paper, we analyze the features of this kind of malware and classify a (non-exhaustive) set of samples of POS RAM scraping malware identified by numerous security vendors [11, 25, 52] regarding these features. We then discuss the evolution of POS RAM scraping malware and also categorize them according to three key aspects: (i) *what is their functionality and what type of persistence is used?*; ii) *how are the processes searched and what (and how) data are scrapped?*; and (iii) *how are the scrapped data exfiltrated from the systems?*. Our results show an evolution in the methods to scrap and exfiltrate sensitive data, aimed at avoiding common detection mechanisms performed by intrusion-detection and software anti-virus solutions, but not so mature as expected. Few families protect their malware samples, and less than a few follows a different behavioral pattern apart from *searching processes on execution*, *opening them*, and *reading its allocated memory*. Surprisingly, we only found two malware families that use anonymous communication as exfiltration channel.

The rest of this paper is as follows. Section 2 reviews the transaction flow of a payment card in a POS system and places where sensitive customer data remain. Then, Section 3 describes data contained in payment cards, regarding different card technologies (namely, magnetic stripe cards, chip cards, and contactless cards). Features of POS RAM scraping malware are introduced in Section 4, while categorizations of 22 samples of this kind of malware from different families, according to the three aforementioned key aspects, are detailed in Section 5. Section 6 provides some background in related works. Finally, conclusions and future work are drawn in Section 7.

2 Point-Of-Sale Card Transaction Flow

This section introduces the transaction flow of a card payment performed in a POS system and provides some notes on industry standards regarding security of payment card architectures.

Figure 1 depicts an abstract view of the transaction flow of a card payment (with on-line verification) as summarized in [16]. A customer selects a card for payment and its cardholder data is entered into the merchant's payment system through a POS terminal. These data are sent to the acquirer bank (merchant's bank) that routes data for processing to the card payment

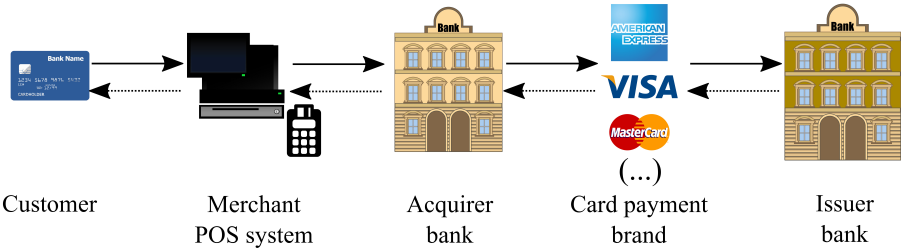


Fig. 1 An abstraction of Point-of-Sale card transaction flow (adapted from [16]).

brand (e.g., VISA, MasterCard, or American Express). Finally, the issuer bank verifies the card is legitimate, not reported as stolen or lost, and whether the customer's account has enough funds/credit available to pay for the transaction. If so, the issuer bank generates an authorization number and routed it back to the card brand which forwards it to the acquirer bank. The acquirer bank then forwards it to the merchant which concludes the sale with the customer, providing her with an acknowledgement (normally in terms of a receipt).

The payment system architecture can be more elaborated, having specific payment gateway, payment switches, and payment data centers. The reader is referred to [20], where several architectures of payment card systems and their intrinsic security are reviewed in detail.

In these scenarios, there exist four key vulnerabilities regarding where sensitive customer data may be accessed. Three of them are related to where cardholder data may stay [20]: (i) *in memory*: data manipulations are carried out by the payment application when processing an authorization or a settlement, and thus, payment card data remain in memory of the processing machine; (ii) *at rest*, when the payment application stores data, either temporarily or for long term, on a disk device; or (iii) *in transit*, when payment data are received and sent to and from other application and devices within the system. The last vulnerability is the application running into the POS system or the payment servers that belong to the merchant, before reaching the acquirer bank: these systems can be compromised (for instance, by an insider technician), modifying the application code that handle customer's sensitive data, or its configuration, to exfiltrate or to change them directly on-the-fly.

In this paper, we focus on malware that target Windows-based POS systems (they represent the 88% of current POS systems [3]) and look for payment-related data that stay in memory.

2.1 A Note on Payment Card Industry Standards

Payment Card Industry (PCI) standards regulate the security of electronic payment systems. These standards apply to different aspects and key players of the electronic payment ecosystem but with a common goal of improving se-

curity. In this context, two standards are relevant: PCI Data Security Standard (PCI DSS), which specifies how sensitive cardholder data must be protected by the merchants and service providers (acquirer/issuer banks); and Payment Application Data Security Standard (PA-DSS), which defines software requirements to be fulfilled by payment applications in compliance with PCI DSS.

Finally, let us remark that the existence of a POS system in compliance with PCI DSS and PA-DSS does not imply the system is secure, since all key vulnerabilities are not fully covered by those PCI requirements. In fact, only data at rest are significantly protected whether the software vendor uses strong cryptographic mechanisms [20].

3 Accessing into Credit Card Data

Data stored in modern credit cards are accessed by four different interfaces: by physical access, by the magnetic stripe, by a chip reader, or by a Near-Field Communication (NFC) reader. Note that the usability of these interfaces strongly depend on the bank card issuer, the card manufacturer, and the POS terminal. For instance, chip cards are not mainstream at the moment in the US, being physical or magnetic stripe data the most common ways of payment.

3.1 Physical Data

Data provided by physical access to the card are widely known since any customer paying goods or services on the Internet have used them. Namely, these data are the following:

- **Name**: cardholder's name, limited up to 26 characters long.
- **Expiration date**: "YY/MM" (year/month) format in the US cards ("MM/YY" format in the EU cards), it specifies the date up to a card is valid to perform any transaction.
- **Credit Card Number**, or Primary Account Number (PAN): defined in [31], it is a 16 to 19-digit number to uniquely identify the card issuer and the account number of the cardholder. First six digits define the issuer identification number (IIN), in which the leading digit defines the major industry identifier (e.g., airlines, banking and financial, or travel and entertainment, among others). Each card issuer is identified by a unique IIN. For instance, VISA cards start by 4, MasterCard cards by 51 (or 55), and American Express cards by 34 (or 37). The next digits define the individual account number of the cardholder. This number has a length in the range 9 to 12 digits. The final digit is a check digit calculated using a checksum formula used to validate the PAN.
- **Card Verification Value** (CVV/CVV2): also known as Card Security Code or Card Identification Number (depends on the card manufacturer), this 3 or 4-digit length security number printed normally on the backside of the card helps to prevent against fraud since it proves the customer has physical access to the card.

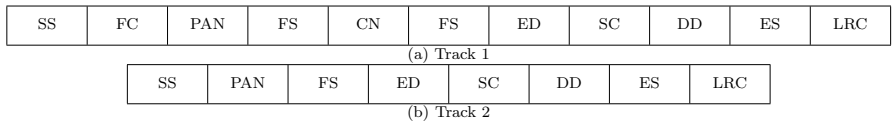


Fig. 2 Diagrams for data contained in magnetic stripe card tracks.

3.2 Magnetic Stripe Cards

Card magnetic stripe, located on the back, is horizontally divided into three tracks. Track 1 and Track 2 contain similar data but different formatted, both standardized in ISO/IEC 7813 [29]. On the contrary, Track 3 is often empty since it is unused by the worldwide networks, and it is standardized in ISO/IEC 4909 [28]. Track 3, also called called THRIFT, was originally intended for use with Automatic Teller Machines. Namely, it was used in Germany as the primary source of authorization for debit card processing. In the following, we only describe Track 1 and Track 2 data in detail since Track 3 is not longer used.

Figure 2(a) summarizes data contained in Track 1. Track 1 format was established by the International Air Transport Association and uses an alpha encoding of 6 bits of data plus 1 odd parity bit per character. The first character is the start sentinel (SS) "%". Then, the next character defines the format code (FC). Herein, we focus on bank format code identified by "B" or "b". Others format code are available, but reserved for proprietary use or individual card issuers. The next characters compose the PAN. Recall that its length is up to 19 characters. Then, the character "^" is used as field separator (FS). The cardholder's name (CN) follows, from 2 to 26 characters long. When this field is unused, the content is an space followed by a surname separator ("/"). Note that MasterCard cards also include right before CN the country code (CC), adding an extra 3 characters [27]. Expiration date (ED) follows the American format of 4 digits "YYMM". Although this field is mandatory for MasterCard and VISA, an FS appears when is unused. The next 3 digits define the service code (SC). These digits determine the interchange and technology (most significant digit), the authorization processing, and range of services and PIN requirements (least significant digit). For instance, a value 2 in the last digit means that the card is valid for paying for goods and services only (i.e., no cash withdrawals allowed). As previous field, this field is mandatory for MasterCard and VISA and an FS appears when is unused. Discretionary Data (DD) appear next. These characters are reserved for proprietary use of card issuer and may include the CVV, the PIN Verification Value (PVV), or the PIN Verification Key Indicator. PVV is a 5-digit length field required by MasterCard and VISA. Finally, the end sentinel (ES) "?" and the Longitude Redundancy Check (LRC) of 1 character long appear.

Data in Track 2 is shorter, as depicted in Figure 2(b). Track 2 format was established by the American Bankers Association and uses a BCD codification of 4 bits of data plus 1 odd parity bit per character. After the SS (a ";" in this

case), the PAN appears. In this format, the character of FS is "=". Before ED, it may appear the CC if PAN starts with 59 (MasterCard cards). As before, ED and SC are mandatory for VISA and MasterCard cards; when unused, an FS will be in place. After these fields, DD appears. The ES character is "?" in this case. Finally, LRC closes Track 2 data.

3.3 Chip Cards

Chip cards, also known as Chip-and-PIN or EMV cards, are the most common credit/debit cards deployed in the EU. Unlike the US, plastic payments in the EU using magnetic stripe are extremely rare. Chip cards were introduced by EMV (stands for Europay, MasterCard and Visa) as a way to authenticate chip-card transactions and minimize the magnetic stripe card counterfeiting fraud. This technology increases the difficulty to replicate the card data, making much harder for criminals to successfully profit from stolen cards.

Unlike magnetic stripe cards, every time an EMV card is used for payment, the chip creates a unique transaction identification that cannot be used again. Besides, instead of swiping the card, it is inserted in a terminal slot where some processing is completed, while it asks for a PIN to authorize the transaction. Thus, any transaction is *theoretically* authorized prior to charge the customer's bank account.

EMV currently leads the debit and credit card payments around Africa, the Middle East, Canada, Latin America, the Caribbean, and Europe as reported in last EMVCo 2014 report [13]. US adoption rate, however, is less than 8%.

EMV protocol presents numerous flaws as pointed out by many works in the literature [1, 4, 5, 40, 41, 47]. PIN and card data can be eavesdropped in EMV transactions by using a hacked terminal, a skimming terminal, or a camera plus double-swipe, among other devices [1]. Similarly, EMV protocol lacks for mechanisms to identify the authenticity of a POS terminal. Additionally, the nonce required to uniquely identified transactions is predictable and generated by the POS terminal (that is, an unreliable party) [4, 5]. Furthermore, EMV protocol (specification version 4) enabled to conduct transactions without formally verifying the PIN, but creating physical evidences (i.e., receipts) as being used, which in fact supposed a real problem for bank customers in fraud disputes [40]. Solutions to improve this system and provide robust evidences are proposed in [41]. These flaws of EMV protocol were proved using formal analysis with F# and ProVerify in [47].

All these problems are argued as the basis for the late adoption in the US [8]. Although there were exist plans for fully deployment of chip-and-PIN cards at the end of 2015, skepticism to adopt EMV protocol in the US has been recommended until it is demonstrated to be fully secure [5]. Nevertheless, the (almost) 13 years of EMV deployed in the EU can serve as lessons learned for the US [2].

Regarding PCI standards and EMV, let us remark that EMV was not designed to protect the confidentiality of sensitive payment data, but as way

to counterfeiting card payment fraud and thus, payment systems still should comply with PCI/PA-DSS, as stated by PCI council in 2010 [44].

3.4 Contactless Cards

The last interface that enables communication with payment cards is based in NFC technology. NFC is a bidirectional short-range (less than 10 cm) contactless communication technology operating in the 13.56 MHz spectrum, based on two Radio Frequency Identification (RFID) standards. Namely, contactless payment cards follow ISO-14443 [30] standard.

Among other applications, the cashless payment sector is where NFC generated more interest [38, 42]. EMV started to deploy contactless payment cards since 2007, expecting NFC to become the major payment scheme in the next years. Recent market research expect to reach more than 500 million of NFC payment users by 2019 [32]. In our opinion, NFC is the natural evolution path of contactless payment methods. Contacless payments are limited to small amounts (e.g., £10, 20€, or US$20). Bigger amounts need the customer to introduce the PIN to validate the transaction. Similarly, the number of consecutive contactless payments without PIN is limited.

NFC security is controversial. For instance, any NFC reader can communicate with a contacless card (that is, there is no identification of the reader). When communication is established, the *contactless track* [48] of a card transmits certain sensitive customer data. Data transmitted include the PAN, cardholder's name, card application ID (contained within the chip and used to communicate with the card by a terminal), card type, and PIN try left. Furthermore, a transaction history is returned. Since communication occurs without knowledge of the cardholder and without checking the identity of the reader, it supposes a high concern with respect to cardholder privacy. Although NFC is based on a proximity concern that theoretically avoids to read contacless cards in large distances, customer's data may be easily harvested for commercial purposes when legitimately paying for good or services in a POS terminal.

Moreover, NFC faces with other potential security threats such as eavesdropping, data modification (i.e., alteration, insertion, or destruction), and relay attacks [7, 22, 23, 39]. Several solutions are proposed to minimize these threats, such as cryptography, distance-bounding protocols, or even particular EMV protocol modifications [17, 22, 23]. Regarding malicious software, malware that feature NFC is theoretically feasible [14, 46, 57] but at the moment there exist no evidences of this kind of malware in-the-wild. However, given the increasing trend of mobile devices with a built-in NFC chip and of mobile malware [51], this situation may change before long.

Fig. 3 Flowchart of stages of a POS RAM scrapping malware sample.

4 Features of POS RAM Scraping Malware

This section introduces our selected features of POS RAM scraping malware, sketched in Figure 4. Recall that these malware aim at stealing customer's payment card data to exfiltrate them and thus perform payment card fraud by several means [56].

In particular, Figure 3 depicts an flowchart of how POS RAM scraping malware work. We mainly distinguish three stages: i) in *Infection & persistence stage*, malware first gain access into a POS system by means of exploiting vulnerabilities or using social-engineering tricks, and later may make itself persistent in the POS system for long-term control and exfiltration. Persistence is normally achieved in Windows OS writing registry keys to initiate malware upon each OS execution; ii) in *Process & data search* stage, malicious activity starts by retrieving the list of process on execution. Each process is then discarded or selected (depending on the type of search performed). For each selected process, its allocated memory is scrapped looking for memory patterns that match card data format; iii) When found, the *Exfiltration* stage begins. In this stage, those data are exfiltrated to the attacker by several means. We have identified different features regarding each stage. In the following, we review these features in detail.

Binary Protection

Malware are statically analyzed when the sample is not executed and all possible execution paths are explored; or dynamically, when the behavior of a sample is analyzed by executing it and monitoring the interaction within its environment (e.g., interaction with file system, registry, and network). Each analysis method has, however, its strengthens and weaknesses [37].

The vast majority of anti-virus software relies on static binary analysis (using signature-based techniques) to detect malware. Thus, a common way to hamper analysis and thus classified a malicious software as benign software is to obfuscate the binary code by using software packers, junk code, or other protection mechanisms [6, 9, 15, 34, 43]. Similarly, a common way to bypass dynamic analysis is to check whether an analysis environment is found. If so, the malware perform no malicious activity or not execute at all. As protection mechanisms, we distinguish between the following: *obfuscation*, *time-driven*, *tampering* (e.g., using checksums mechanisms), *VM execution*, *encryption* (e.g., packed), *time-driven* (e.g., malicious activity is triggered based on time events or at specific time); and *anti-debugging* (e.g., the sample incorporates code

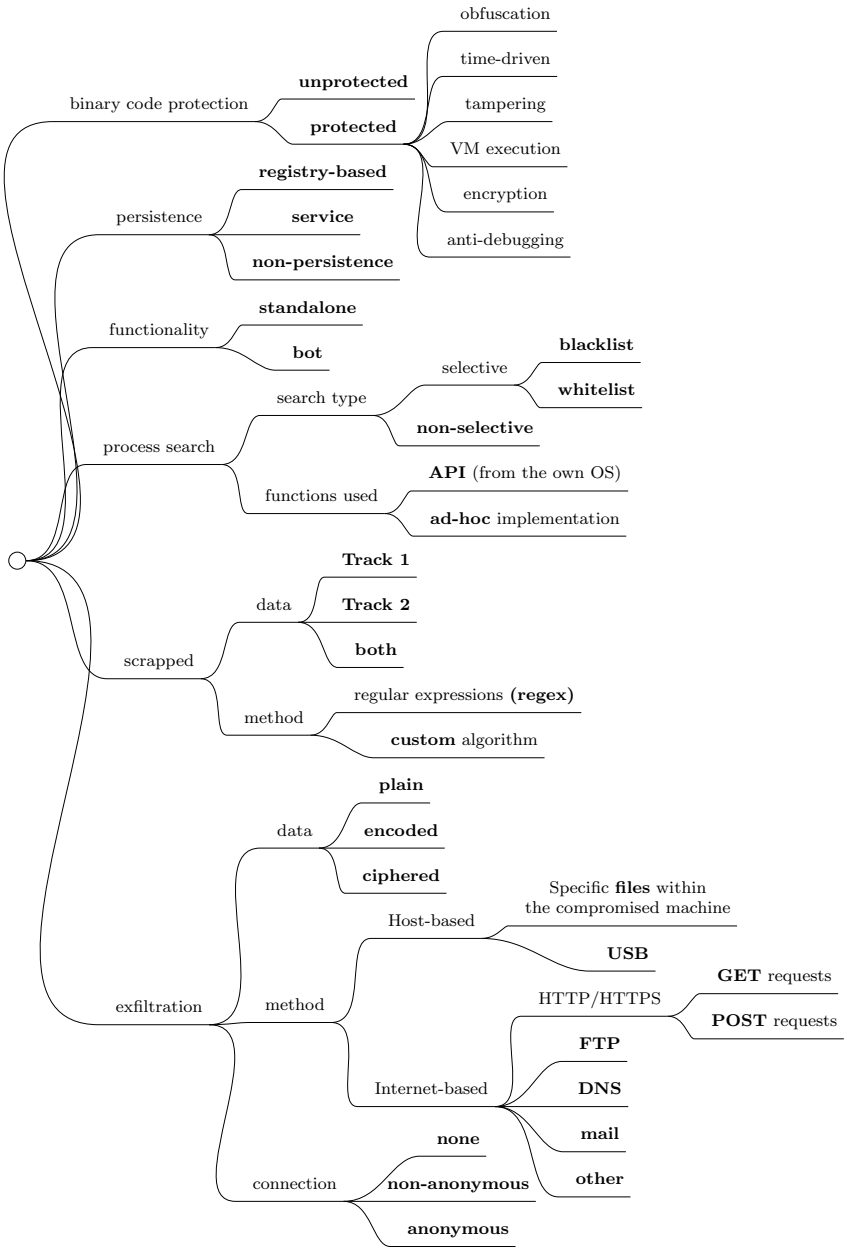


Fig. 4 Features of POS RAM scraping malware.

to hamper dynamic analysis). Since the samples under analysis in this paper mainly present some sort of encryption (namely, using software packers), we consider only **protected** and **unprotected** as a coarse-grained feature of selected samples.

Persistence

This category relates how malware operate to become persistent in the compromised machine [59] (i.e., to execute after each startup of the compromised machine). Malware can persist by means of the Windows registry by creating specific autorun keys. Similarly, it can persist as an OS service. In this case, a registry key is also created but malware are executed as a service instead of a normal process. In the same manner, malware are non-persistent when they are not automatically executed after each startup.

Functionality

POS RAM scraping malware have bot functionality when communicate with command-and-control (C&C) servers to receive orders/send responses (for instance, to download and execute other malware, to update, or even to kill itself); otherwise, they have standalone functionality (i.e., they only scan memory of processes to find out and exfiltrate sensitive card data).

Process Search Type

Once installed in the system, the malware collect all processes on execution. Processes are then filtered to explore their allocated memory. The candidate processes to be explored are chosen following different criteria: *non-selective*, when all processes are explored indiscriminately, or *selective*, when only specific processes are taken. The list of selective processes can be optionally in (that is, *whitelisting*) or optionally out (that is, blacklisting). To this regard, the existence of whitelist-selective POS malware in a system may indicate a previous knowledge of the system being attacked and hence, a high likelihood of having a system under an Advanced Persistent Threat (APT) attack.

Process Search Function Used

The Windows structure `PROCESSENTRY32` describes an entry from a list of processes residing in the system address space when a snapshot of processes on execution was taken[1]. This structure is handled by some Windows API

[1]See `https://msdn.microsoft.com/en-us/library/windows/desktop/ms684839(v=vs.85).aspx` for details.

functions to iterate on the list of processes. Examples of these APIs are `CreateToolhelp32Snapshot`, `EnumProcesses`, or non-documented APIs as `ZwQuerySystemInformation`. POS malware commonly use these APIs to handle the list of processes on execution. Hence, a good indicator of malware behavior is to monitor the use of these functions. Instead of using those functions, POS malware may use their ad-hoc functions as a way to thwart detection.

Scrapped Data

Sensitive data, as described in Section 3.2, are contained in Track 1 and Track 2. POS RAM scraping malware can look for either Track 1, Track 2, or both data within the boundaries of a process memory.

Scrapped Method

The method to look for memory patterns that match Track 1 or Track 2 data can be implemented by regular expressions or other code structures such as byte-to-byte comparison loops. Some samples analyzed in this paper incorporate regular expressions as matching method, as described in Section 5.1.

Exfiltrated Data

Once POS RAM scraping malware detect sensitive data of interest, cybercriminals need to exfiltrate those data from the system to commit fraud. These data is exfiltrated as plain, encoded (e.g., using algorithms such as base32, base64, or base85, among others), or ciphered text. Cybercriminals tend to ciphering to avoid detection performed by network monitoring tools.

Exfiltration Method

Different exfiltration methods have been observed. There exist POS malware that perform a *host-based* exfiltration instead of connecting to a C&C server. That is, they dump scrapped data into specific files in the own disk of the compromised system, into shared network folders, or directly into specific USB devices. The latter case indicates the attacker somehow has physical access to the compromised system. Other POS malware connect through the *Internet* by different means to complete the data extraction. To this regard, we have observed HTTP(S) GET/POST requests, FTP requests, and DNS requests, among others.

Exfiltration Connection

We have also distinguished the connection performed (if any) to exfiltrate the sensitive data. Thus, exfiltration connection can be anonymous when using TOR network or non-anonymous otherwise. Up to date and to the best of our knowledge, only two samples of different malware families connect to the Internet anonymously.

5 Classification of Malware Samples and Discussions

In this section, we review a (non-exhaustive) list of POS malware families according to our selected features. We collected a total set of 144 malware samples that target Windows-based POS systems from a private malware repository, research forums, and individual security researchers after reviewing some industry reports [11, 25]. All these samples are available for freely downloading at `http://webdiis.unizar.es/~ricardo/pos-ram-scraping-malware-samples`.

We upload each sample to VirusTotal (VT) website, which automatically checks the submitted file using anti-virus engines from multiple software vendors. For each malware family, we selected the sample with highest detection ratio in VT as the most representative sample of the family. We assume that the more number of anti-virus engines detect a sample, the more indicators of malicious behavior a sample includes. Then, we classify individual samples of each POS RAM scraping malware family considered in this paper. As future work, we aim at further studying samples deeply regarding their code morphology to determine whether characterization of a single instance can be extrapolated to characterize the malware family.

Table 1 shows the MD5 hash of the selected sample and its VT detection ratio (in January 14, 2016) of malware families considered in this paper, ordered by discovery date. We divided each year in quarters since it is difficult to determine a concrete discovery date. Each of those selected samples are statically analyzed in a virtual machine running Windows 7 Professional SP1 using reverse engineering tools such as radare2, PEiD, and IDA Pro disassembler. Dynamic analysis is carried out using a web service that runs a well-known dynamic malware analysis tool (namely, Cuckoo Sandbox). In some doubtful cases, we have performed a dynamic analysis using OllyDBG. Furthermore, we have also verified the presence of regular expressions in all malware samples collected.

We first discuss malware evolution, according to types of persistence, and then we classify each sample, according to the three stages of its malicious activity (*infection & persistence*, *process & data search*, and *exfiltration*).

Malware family	Other names	Discovery date	Selected sample	VT ratio
rdasrv		2011 (Q4)	516cef2625a822a253b89b9ef523ba37	47 out of 52
ALINA		2012 (Q4)	1efeb85c8ec2c07dc0517ccca7e8d743	46 out of 55
Dexter		2012 (Q4)	70feec581cd97454a74a0d7c1d3183d1	50 out of 54
vSkimmer		2013 (Q1)	dae375687c520e06cb159887a37141bf	48 out of 55
BlackPOS	KAPTOXA, Reedum	2013 (Q2)	d9cc74f36ff173343c6c7e9b4db228cd	45 out of 52
FYSNA	Chewbacca	2013 (Q4)	21f8b9d9a6fa3a0cd3a3f0644636bf09	47 out of 55
Decebal		2014 (Q1)	d870d85e89f3596a016fdd393f5a8b39	41 out of 55
JackPOS		2014 (Q1)	75990dde85fa2722771bac1784447f39	41 out of 52
Soraya		2014 (Q2)	1483d0682f72dfefff522ac726d22256	43 out of 55
BackOff	PoSeidon, FindPOS	2014 (Q3)	17e1173f6fc7e920405f8dbde8c9ecac	49 out of 56
BrutPOS		2014 (Q3)	95b13cd79621931288bd8a8614c8483f	42 out of 53
FrameworkPOS	BlackPOS v2	2014 (Q3)	b57c5b49dab6bbd9f4c464d396414685	45 out of 56
GetmypassPOS		2014 (Q4)	1d8fd13c890060464019c0f07b928b1a	35 out of 56
LusyPOS		2014 (Q4)	bc7bf2584e3b039155265642268c94c7	47 out of 56
LogPOS		2015 (Q1)	af13e7583ed1b27c4ae219e344a37e2b	44 out of 56
Punkey		2015 (Q2)	b1fe4120e3b38784f9fe57f6bb154517	44 out of 56
FighterPOS		2015 (Q2)	b0416d389b0b59776fe4c4ddeb407239	43 out of 57
NitlovePOS		2015 (Q2)	6cdd93dcb1c54a4e2b036d2e13b51216	47 out of 56
MalumPOS		2015 (Q2)	acdd2cffc40d73fdc11eb38954348612	36 out of 56
BernhardPOS		2015 (Q3)	e49820ef02ba5308ff84e4c8c12e7c3d	43 out of 56
GamaPOS		2015 (Q3)	58e5dd98015164b40de533e379ed6ac8	43 out of 55
AbbaddonPOS		2015 (Q4)	46810f106dbaaff5c3c701c71aa16ee9	39 out of 56

Table 1 List of POS RAM scraping malware samples selected.

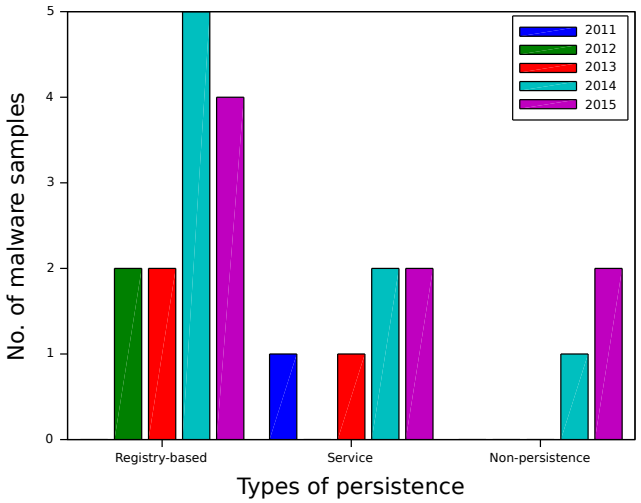
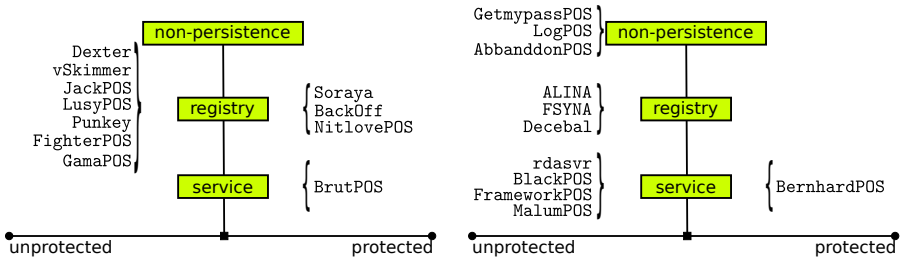


Fig. 5 Evolution of POS RAM scraping malware from 2009 to 2015.

5.1 Discussion

5.1.1 On Evolution

First, we discuss the evolution of this kind of malware. Regarding discovery date, a rapidly increasing of POS RAM scraping malware families is observed. Malware developers usually deploy their samples in Windows-based environments [33] and hence, to get advantage of this knowledge in the domain of POS systems is easy since almost 88% of them are based on Windows (in different flavors) [3].



(a) Bot functionality (b) Standalone functionality

Fig. 6 Categorization of POS RAM scrapping malware samples based on binary protection, persistence method, and functionality.

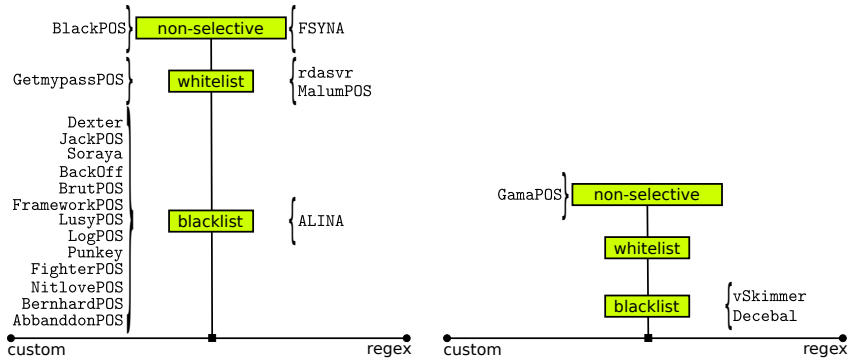
Figure 5 depicts the evolution of POS RAM scraping malware from 2009 to 2015, according to the type of persistence. It is remarkable that at least two new POS RAM scraping malware families appear per year from 2012 onwards, in case of registry-based persistence. Similarly, it also noticeable that malware using other types of persistence starts appearing from 2014 onwards. We believe that the number of those samples, specially using service-based persistence, will start increasing in the following years.

5.1.2 On Infection and Persistence

Figure 6 depicts a classification of samples of POS RAM scraping malware families based on binary protection, persistence method, and functionality.

First, we discuss the protection mechanisms. As tools, we have used PEiD and RDG Packer Detector, since we aim at knowing the compiler or the packer used in the binary. Although nowadays malware are typically deployed using some sort of protection mechanism as an attempt to evade detection [21, 61], only 5 out of 22 malware families analyzed present some sort of protection (that is, packing). We observe the use of UPX (a well-known packer) and of a custom packer. Most of samples analyzed are developed with C++ or Delphi. `GamaPOS` is the first POS malware seen in-the-wild developed with .NET framework. We believe the number of incidents targeting POS system will follow this trend, increasing in the years to come. Similarly, we believe malware samples will be distributed in complex obfuscation forms such as custom packers, as well as adding advanced anti-analysis tricks (we detected only 3 families with this kind of tricks at the moment).

Regarding persistence, most of samples present registry-based persistence. Surprisingly, 3 malware families present no persistence mechanism. This fact indicate that these samples could be executed as second (or further) stage of other attacks. Only 6 families use a service-based persistence. Note that service-based persistence increases the difficulty to detect strange activity process, since Windows services are running in background (normally as an instance of the `svchost.exe`) and not interacting with the desktop. An interest-



(a) Both tracks (b) Track 2

Fig. 7 Categorization of POS RAM scrapping malware samples based on process search, scrapped data, and scrapped method.

ing persistence behavior is shown by `NitlovePOS` sample, since it uses NTFS alternate data streams to hide into the system.

Regarding functionality, almost half of malware families incorporate botnet-based functionality. To this regard, it is worth mentioning `GamaPOS` that is indeed part of the Andromeda botnet [62].

We believe the number of POS malware with bot functionality, protected, and service-based persistence (or using other anti-forensic techniques as `NitlovePOS` does) will increase in the future, since these features allow to easily maintain control of a compromised machine installing updates or other pieces of malware such as keyloggers, to hamper reverse engineering analysis, and to stay longer undetected into the compromised system. Hence, to keep software updated and to deploy intrusion-detection systems may help to detect and mitigate these threats.

5.1.3 On Process and Data Search

Figure 7 depicts a classification of samples of POS RAM scraping malware families, based on process search, scrapped data, and scrapped method.

Regarding process search, most of the samples use a blacklist approach to search processes of interest. `AbbaddonPOS` behaves differently, since it only excludes itself from memory processing. Only 3 malware samples perform a search focused on particular processes and with interest in both tracks, using different scrapped methods. The same number performs a non-selective process search.

Surprisingly, all malware samples considered in this paper use Windows APIs to collect the list of running processes. Most of them use `CreateToolhelp32Snapshot` function, while others use `EnumProcesses`. Only one sample uses the not so-common `ZwQuerySystemInformation` function to this aim (namely, `BernhardPOS`. Lastly, it is worth mentioning that all samples included in this paper, but `LogPOS` and `BernhardPOS`, use the same approach:

first, they iterate on the running processes; then, processes of interest are opened as a handle; and finally, they read the mapped memory of each process (normally by memory chunks of specific size). `LogPOS` and `BernhardPOS` perform in a different manner since the function to read and scrap the memory is directly injected to each process independently. That is, the search function is performed from the own process' memory space. We believe this search is performed in this way to avoid detection of process monitoring tools. To overcome the use of these APIs for process listing, malware require to dispose a rootkit or driver functionality to easily scrape every memory page currently mapped in the system. At the moment, and to the best of our knowledge, there exists no POS RAM scraping malware family with this functionality – however, we expect to discover them before long.

Regarding scrapped data, most of the samples use custom algorithms to look for data matching card data patterns. Note that these methods may produce a high number of false positives. To this regard, it is worth mentioning some samples include a custom implementation of Luhn algorithm, which is the checksum formula used to validate credit card numbers specified in ISO/IEC 7812-1 [31]. Namely, these samples are `Dexter`, `Decebal`, `Soraya`, `BernhardPOS`, `BrutPOS`, `GetmypassPOS`, `LogPOS`, `Punkey`, and `AbbaddonPOS`.

Finally, data of interest are either both tracks or only Track 2. That is, no malware look for only Track 1 data. Most of the malware samples, indeed, look for both tracks (upper-side in Figure 7). Regular expressions are only found in samples of the earliest discovered malware families (with the exception of `MalumPOS`). The reason behind this is obvious, since regular expressions are hardcoded into the file, and then they are very easy to find. Thus, a binary that incorporates regular expressions hardcoded is rapidly detected as malware. Figures 8 and 9 show the regular expressions found on some of the samples analyzed (namely, in `rdasrv` samples). The regular expression of Figure 8 matches Track 1 of cards with expiration date ranging from 2007 to 2015. Similarly, the expression of Figure 9 matches Track 2 of cards whose first PAN number is between 3 to 9 (e.g., VISA, MasterCard, or JCB payment cards, among others) and expiring between 2011 to 2015.

In our opinion, the number of samples doing a blacklist search and using custom methods for scrapping data will keep increasing in the future, since these methods allow to attack indiscriminately to any system and to have a low likelihood of detection by static signature methods. Similarly, the interest in both tracks will also prevail since the likelihood to perform a successful fraud is increased.

Let us also remark that a good defense method against these malware is to monitor the APIs for process listing. Given that malware need rootkit or driver functionality to avoid the use of these functions, to deploy monitoring software of these APIs would increase the likelihood detection of POS RAM scraping malware.

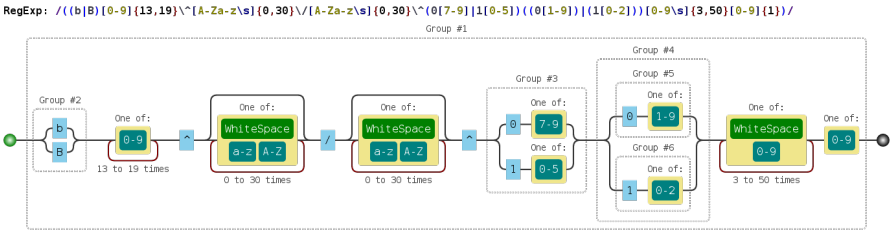


Fig. 8 Regular expression found hardcoded in binary data of some malware samples to match Track 1 data. Group #3 ranges expiration date scrapped (from 2007 to 2015, in this case).

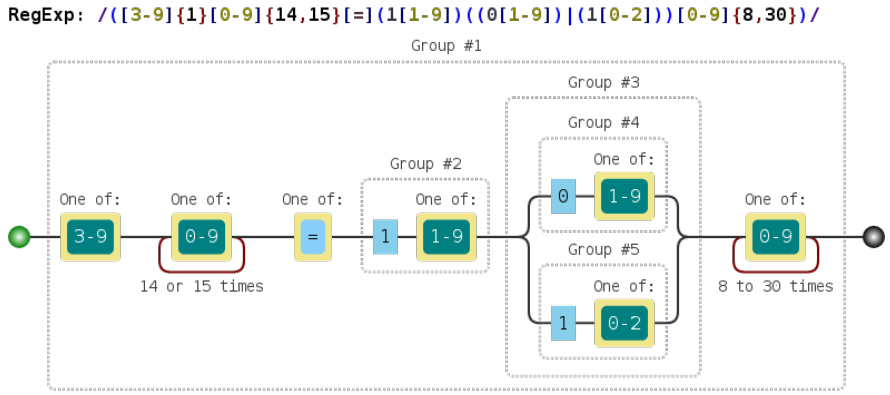
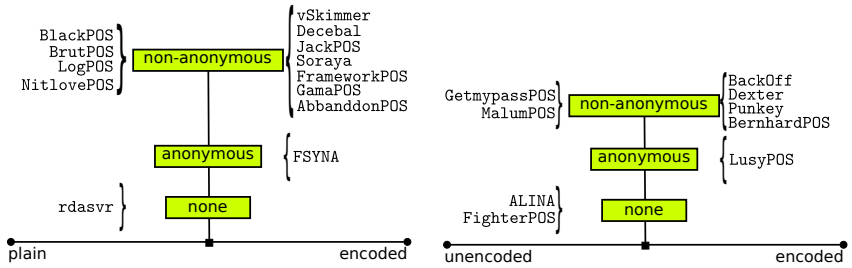


Fig. 9 Regular expression found hardcoded in binary data of some malware samples. Valid for Track 2 data having first PAN digit ranging from 3 to 9 and expiration date between 2011 to 2019 (group #2).

5.1.4 On Exfiltration

Figure 10 depicts a classification of samples of POS RAM scraping malware families, based on how data are exfiltrated and the exfiltration communication.

Almost all samples exfiltrate scrapped data encoded or/and ciphered to avoid network packet-level detection algorithms. Regarding exfiltration methods, they commonly use HTTP POST to specific servers (normally hardcoded into the binary code) to exfiltrate sensitive data. Only 3 samples generate a file inside the compromised machine with the scrapped data. That is, they need other ways to exfiltrate them. These samples are, namely, `rdasrv`, `GetmypassPOS`, and `MalumPOS`. Of course, other variants of these families may implement alternative exfiltration methods. Other strange exfiltration methods observed in this study include the use of DNS requests or the use of specific volume names in USB drives, such as `vSkimmer` does. The latter method clearly indicates that the attacker has physical access into the machine. To this regard, it is worth also mentioning `AbbaddonPOS`, which implements an own protocol for exfiltration.



(a) Non-ciphered (b) Ciphered

Fig. 10 Categorization of POS RAM scrapping malware samples based on exfiltration.

Finally, most of samples use a non-anonymous connection to a server where data are exfiltrated, which makes easier to track the attacker, identify the origin, and take down the infrastructure of the attacker. Surprisingly, two samples use an anonymous connection using TOR network. In this case, the task to track, identify, and take down the infrastructure becomes almost impossible.

Unfortunately, we believe the use of TOR-based communication will become a common feature of future POS RAM scraping malware. The deployment of network-monitoring tools to detect and prevent anonymous connections may help to mitigate these threats.

Our findings show that few malware families behave exactly equal during all stages. Otherwise, some samples would be attributed to a single family instead of to different ones. Some features stand out as the most used by this kind of malware, such as registry-based persistence, searching process based on blacklist, seeking data using custom match functions that comply with both Track 1 and Track 2 data format, and communicating throughout non-anonymous channels to exfiltrate non-ciphered but encoded data. As future work, we aim at analyzing the collected samples in more detail to discover relationship among families. Metrics as binary-code reusing and behavioral patterns based on call-graph approaches may help us also to fulfill this goal.

6 Related Work

Several taxonomies have been proposed aimed at specific malware types. However, to the best of our knowledge, we are the first to propose a comprehensive classification of POS RAM scraping malware according to their behavior. In the following, we review some of these other taxonomies. A taxonomy of computer worms, a kind of malware that self-propagate through a network, was presented in [60]. Similarly, in [36] a taxonomy of advanced persistent threats was proposed. A taxonomy of a kind of malware that detect when are being analyzed and then behave in a non-malicious manner was presented in [35]. A taxonomy of botnet structures based on their utility to the botmaster was proposed in [10]. Likewise, a taxonomy of software packers (normally employed

to obfuscate malware when deployed) based on their run-time complexity is proposed in [55].

Regarding POS systems vulnerabilities, solutions such as real-time monitoring to detect anomalies using invariant mining techniques or fault injection testing have been proposed [49]. In [50], a deep analysis of POS system security was performed and solutions to improve their security were provided. For instance, to use a firewall, to restrict POS terminals to operate with least privileges as needed, and to harden the OS within the POS system were proposed, among other recommendations.

In [24], a tool was proposed to automatically identify the credit card data flow in commercial payment systems running on cloud-based servers by inspecting data at network level (a kind of data scraping). Lastly, it is worth mentioning [18] that provided a security analysis of different audio-magnetic strip readers. The authors showed how a crafted mobile application might access to cardholder data from these devices.

7 Conclusions

Connected POS systems are becoming the preferred target for cybercriminals. When customers pay merchants for food or services, their sensitive payment card data are maintained in different locations, from where data may be retrieved and exfiltrated to commit fraud. Nowadays, RAM scraping is the major threat that connected POS systems face to. Thus, cybercriminals craft pieces of malware to scan memory of processes being executed on these systems, to search payment-related data and to exfiltrate these data through the system boundaries. Most of the largest data breach occurred in last years to US companies (such as Target, Home Depot, or Staples) were caused by malware that feature RAM scraping into POS systems.

In this paper, we studied the evolution of POS RAM scraping malware and extracted their features based on how they behave in a POS system when looking for card data that stay in memory. In particular, we considered three different stages of behavior: infection and persistence, process and card data search, and data exfiltration. We reviewed and classified samples of 22 well-known POS RAM scraping malware families according to these features. Our results shown that evolution of these malware is still immature: few families use techniques to avoid static or dynamic malware analysis (e..g, software packing, code obfuscation, or analysis awareness tricks) and they still use detectable methods for persistence (most of them use a registry-based persistence), process and data search (using blacklist search and with interest of both Track 1 and Track 2 card data), and data exfiltration (using non-anonymous channels and encoded data). We found that only two samples of different malware families communicate through the TOR anonymous network, which may thwart to take fraud criminal activity down timely.

As future work, we would like to analyze a larger set of samples of these families to identify similarities between them, thus finding out relationships

among families. Similarly and based on a more fine-grained behavioral analysis, we aim at detecting new samples of these families and better classifying existing ones. We also aim at studying other kind of POS malware that target data in transit or data in rest in POS systems. Other features are also considered to be included, such as code morphology that may help to determine whether specific results in analysis of a sample of a given malware family are extensible to the malware family.

Acknowledgements This work was partially supported by the Spanish MICINN project CyCriSec (TIN2014-58457-R). The author would like to thank Marc Rivero and Rubén Espadas, MLW.RE NPO, for providing malware samples, Xylitol for maintaining the thread in KernelMode forum of POS RAM scraping malware, and the anonymous referees for providing constructive comments and helping to improve the contents of this paper.

References

1. Adida B, Bond M, Clulow J, Lin A, Murdoch S, Anderson R, Rivest R (2009) Phish and Chips. In: Christianson B, Crispo B, Malcolm J, Roe M (eds) Proceedings of the 14th International Workshop on Security Protocols, Springer Berlin Heidelberg, Lecture Notes in Computer Science, vol 5087, pp 40–48, DOI 10.1007/978-3-642-04904-0_7
2. Anderson R, Murdoch SJ (2014) EMV: Why Payment Systems Fail. Commun ACM 57(6):24–28, DOI 10.1145/2602321
3. Bodhani A (2013) Turn on, log in, checkout. Engineering Technology 8(3):60–63, DOI 10.1049/et.2013.0308
4. Bond M, Choudary O, Murdoch S, Skorobogatov S, Anderson R (2014) Chip and Skim: Cloning EMV Cards with the Pre-play Attack. In: IEEE Symposium on Security and Privacy (SP), pp 49–64, DOI 10.1109/SP.2014.11
5. Bond M, Choudary M, Murdoch S, Skorobogatov S, Anderson R (2015) Be Prepared: The EMV Preplay Attack. IEEE Security & Privacy 13(2):56–64, DOI 10.1109/MSP.2015.24
6. Borello JM, Mé L (2008) Code obfuscation techniques for metamorphic viruses. Journal in Computer Virology 4(3):211–220, DOI 10.1007/s11416-008-0084-2
7. Brandt NB, Stamp M (2014) Automating NFC message sending for good and evil. Journal of Computer Virology and Hacking Techniques 10(4):273–297, DOI 10.1007/s11416-014-0223-x
8. Caldwell T (2014) Securing the point of sale. Computer Fraud & Security 2014(12):15–20, DOI http://dx.doi.org/10.1016/S1361-3723(14)70557-3
9. Collberg CS, Thomborson C (2002) Watermarking, Tamper-Proofing, and Obfuscation – Tools for Software Protection. IEEE Trans Soft Eng 28(8):735–746
10. Dagon D, Gu G, Lee C, Lee W (2007) A Taxonomy of Botnet Structures. In: Proceedings of the 23rd Annual Computer Security Applications Conference (ACSAC), pp 325–339, DOI 10.1109/ACSAC.2007.44

11. Dell SecureWorks Counter Threat Unit (2013) Point-of-Sale Malware Threats. Tech. rep., Dell SecureWorks Inc., available at http://www.secureworks.com/cyber-threat-intelligence/threats/point-of-sale-malware-threats/.
12. Department of Homeland Security (2010) National Security Strategy. The White House, available at http://www.whitehouse.gov/sites/default/files/rss_viewer/national_security_strategy.pdf.
13. EMVCo (2015) EMV Card-Present Transaction Percentage. Online, accessed October 25, 2015. https://www.emvco.com/documents/EMVCo_Card_present_EMV.pdf
14. Felt AP, Finifter M, Chin E, Hanna S, Wagner D (2011) A Survey of Mobile Malware in the Wild. In: Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices (SPSM), ACM, New York, NY, USA, pp 3–14, DOI 10.1145/2046614.2046618
15. Filiol E (2007) Metamorphism, Formal Grammars and Undecidable Code Mutation. International Journal of Computer, Electrical, Automation, Control and Information Engineering 1(2):281–286
16. FirstData (2010) Payments 101: Credit and Debit Card Payments – Key Concepts and Industry Issues. [Online], https://www.firstdata.com/en_us/insights/payments-101-white-paper-/_jcr_content/content-block/insight_individual/insights-downloads-par/download/file.res/fd-Payments-101-Credit-and-Debit-Card-Payments-white-paper.pdf
17. Francis L, Hancke G, Mayes K, Markantonakis K (2012) Practical Relay Attack on Contactless Transactions by Using NFC Mobile Phones. In: Lo NW, Li Y (eds) Proceedings of the 2012 Workshop on RFID and IoT Security (RFIDsec 2012 Asia), IOS Press, Cryptology and Information Security Series, vol 8, pp 21–32
18. Frisby W, Moench B, Recht B, Ristenpart T (2012) Security Analysis of Smartphone Point-of-sale Systems. In: Proceedings of the 6th USENIX Conference on Offensive Technologies, USENIX Association, Berkeley, CA, USA, WOOT'12, pp 1–12
19. Gold S (2014) The evolution of payment card fraud. Computer Fraud & Security 2014(3):12–17, DOI http://dx.doi.org/10.1016/S1361-3723(14)70471-3
20. Gomzin S (2014) Hacking Point of Sale: Payment Application Secrets, Threats, and Solutions, 1st edn. John Wiley & Sons Inc.
21. Guo F, Ferrie P, Chiueh Tc (2008) A Study of the Packer Problem and Its Solutions. In: Lippmann R, Kirda E, Trachtenberg A (eds) Recent Advances in Intrusion Detection (RAID), Lecture Notes in Computer Science, vol 5230, Springer Berlin Heidelberg, pp 98–115, DOI 10.1007/978-3-540-87403-4_6
22. Hancke G, Mayes K, Markantonakis K (2009) Confidence in smart token proximity: Relay attacks revisited. Computers & Security 28(7):615–627, DOI http://dx.doi.org/10.1016/j.cose.2009.06.001

23. Haselsteiner E, Breitfuß K (2006) Security in Near Field Communication (NFC) – Strengths and Weaknesses. In: Proceedings of the Workshop on RFID Security and Privacy (RFIDSec)
24. Hizver J, Chiueh Tc (2011) Automated Discovery of Credit Card Data Flow for PCI DSS Compliance. In: Proceedings of the 2011 IEEE 30th International Symposium on Reliable Distributed Systems (SRDS), IEEE Computer Society, Washington, DC, USA, pp 51–58, DOI 10.1109/SRDS.2011.15
25. Huq N (2014) PoS RAM Scraper Malware: Past, Present, and Future. Tech. rep., Trend Micro Inc., available at http://www.trendmicro.com/cloud-content/us/pdfs/security-intelligence/white-papers/wp-pos-ram-scraper-malware.pdf.
26. Huq N (2015) Defending Against PoS RAM Scrapers: Current Strategies and Next-Gen Technologies. Tech. rep., Trend Micro Inc., available at http://www.trendmicro.com/cloud-content/us/pdfs/security-intelligence/white-papers/wp-defending-against-pos-ram-scrapers.pdf.
27. International Organization for Standardization (1997) ISO/IEC 3166-1:1997: Codes for the representation of names of countries and their subdivisions – Part 1: Country codes. URL http://www.iso.org/iso/catalogue_detail?csnumber=24591
28. International Organization for Standardization (2006) ISO/IEC 4909:2006: Identification cards – Financial transaction cards – Magnetic stripe data content for track 3.
URL http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=43309
29. International Organization for Standardization (2006) ISO/IEC 7813:2006: Information technology – Identification cards – Financial transaction cards.
URL http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=43317
30. International Organization for Standardization (2013) ISO/IEC 18092:2013: Information technology – Telecommunications and information exchange between systems – Near Field Communication – Interface and Protocol (NFCIP-1).
URL http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=56692
31. International Organization for Standardization (2015) ISO/IEC 7812-1:2015: Identification cards – Identification of issuers – Part 1: Numbering system.
URL http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=66011
32. Juniper Research Limited (2014) Apple Pay and HCE to Push NFC Payment Users to More Than 500 Million by 2019. [Online; accessed at November 2, 2014],

http://www.marketwired.com/press-release/apple-pay-hce-push-nfc-payment-users-more-than-500-million-2019-juniper-research-finds-1961558.htm
33. Kaspersky Lab (2014) Kaspersky Security Bulletin 2014. [Online], available at http://securelist.com/files/2014/12/Kaspersky-Security-Bulletin-2014-EN.pdf.
34. Lin D, Stamp M (2010) Hunting for undetectable metamorphic viruses. Journal in Computer Virology 7(3):201–214, DOI 10.1007/s11416-010-0148-y
35. Lindorfer M, Kolbitsch C, Milani Comparetti P (2011) Detecting Environment-Sensitive Malware. In: Proceedings of the 14th International Symposium on Recent Advances in Intrusion Detection (RAID), Springer Berlin Heidelberg, Lecture Notes in Computer Science, vol 6961, pp 338–357, DOI 10.1007/978-3-642-23644-0_18
36. Line MB, Zand A, Stringhini G, Kemmerer R (2014) Targeted Attacks Against Industrial Control Systems: Is the Power Industry Prepared? In: Proceedings of the 2nd Workshop on Smart Energy Grid Security (SEGS), ACM, New York, NY, USA, SEGS '14, pp 13–22, DOI 10.1145/2667190.2667192
37. Liu K, Tan HBK, Chen X (2013) Binary Code Analysis. Computer 46(8):60–68, DOI 10.1109/MC.2013.268
38. de Looper C (2015) Mobile Payment Boasts Rosy Future, But Some Obstacles Remain in Play. [Online; accessed at January 23, 2015], http://www.techtimes.com/articles/24762/20150106/mobile-payments-worth-130-billion-2020.htm
39. Mitrokotsa A, Rieback MR, Tanenbaum AS (2010) Classifying RFID attacks and defenses. Information Systems Frontiers 12(5):491–505, DOI 10.1007/s10796-009-9210-z
40. Murdoch S, Drimer S, Anderson R, Bond M (2010) Chip and PIN is Broken. In: IEEE Symposium on Security and Privacy (SP), pp 433–446, DOI 10.1109/SP.2010.33
41. Murdoch SJ, Anderson R (2014) Security Protocols and Evidence: Where Many Payment Systems Fail. In: Christin N, Safavi-Naini R (eds) Proceedings of the 18th International Conference on Financial Cryptography and Data Security (FC), Springer Berlin Heidelberg, Lecture Notes in Computer Science, vol 8437, pp 21–32, DOI 10.1007/978-3-662-45472-5_2
42. Oak C (2015) The year 2014 was a tipping point for NFC payments. [Online; accessed at January 15, 2015], http://www.finextra.com/blogs/fullblog.aspx?blogid=10382
43. Oorschot P (2003) Revisiting Software Protection. In: Boyd C, Mao W (eds) Proceedings of the 6th International Conference on Information Security (ISC), Springer Berlin Heidelberg, Lecture Notes in Computer Science, vol 2851, pp 1–13, DOI 10.1007/10958513_1
44. PCI Security Standards Council (2010) PCI DSS Applicability in an EMV Environment – A Guidance Document. [Online], available at www.

pcisecuritystandards.org/documents/pci_dss_emv.pdf.
45. Rantos K, Markantonakis K (2014) Analysis of Potential Vulnerabilities in Payment Terminals. In: Markantonakis K, Mayes K (eds) Secure Smart Embedded Devices, Platforms and Applications, Springer New York, pp 311–333, DOI 10.1007/978-1-4614-7915-4_13
46. Rieback M, Crispo B, Tanenbaum A (2006) RFID Malware: Truth vs. Myth. IEEE Security & Privacy 4(4):70–72, DOI 10.1109/MSP.2006.102
47. de Ruiter J, Poll E (2012) Formal Analysis of the EMV Protocol Suite. In: Mödersheim S, Palamidessi C (eds) Theory of Security and Applications, Lecture Notes in Computer Science, vol 6993, Springer Berlin Heidelberg, pp 113–129, DOI 10.1007/978-3-642-27375-9_7
48. Sanders R (2008) From EMV to NFC: the contactless trail? Card Technology Today 20(3):12–13, DOI http://dx.doi.org/10.1016/S0965-2590(08)70077-X
49. Sarkar S, Mitra S, Roy A (2014) Point of sale vulnerabilities: Solution approach. Tech. rep., Infosys
50. Smith DC (2014) Preventing point-of-sale system intrusions. Tech. rep., Naval Postgraduate School
51. Suarez-Tangil G, Tapiador J, Peris-Lopez P, Ribagorda A (2014) Evolution, Detection and Analysis of Malware for Smart Devices. IEEE Communications Surveys Tutorials 16(2):961–987, DOI 10.1109/SURV.2013.101613.00077
52. Symantec Security Response (2014) Attacks on point-of-sales systems. Tech. rep., Symantec, available at http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/attacks_on_point_of_sale_systems.pdf.
53. Trend Micro (2014) Point-of-Sale System Breaches: Threats to the Retail and Hospitality Industries. Tech. rep., Trend Micro Inc., available at http://www.trendmicro.com/cloud-content/us/pdfs/security-intelligence/white-papers/wp-pos-system-breaches.pdf.
54. Trustwave (2014) Combatting Point-of-Sale Malware. Tech. rep., Trustware Holdings Inc., available at https://www.trustwave.com/Resources/Library/Documents/Combatting-Point-of-Sale-Malware/.
55. Ugarte-Pedrero X, Balzarotti D, Grueiro IS, Bringas PG (2015) SoK: Deep Packer Inspection: A Longitudinal Study of the Complexity of Run-Time Packers. In: Proceedings of the 36th IEEE Symposium on Security and Privacy, vol PP, p PP
56. Upendar J, Rao EG (2013) An Overview of Plastic Card Frauds and Solutions for Avoiding Fraudster Transactions . International Journal of Research in Engineering and Technology 2(8):215–222
57. Vila J, Rodríguez RJ (2015) Practical Experiences on NFC Relay Attacks with Android: Virtual Pickpocketing Revisited. In: Proceedings of the 11th International Workshop on RFID Security (RFIDsec), Springer, Lecture Notes in Computer Science, vol 9440, pp 87–103, DOI 10.1007/978-3-319-

24837-0_6
58. Walters R (2014) Cyber Attacks on U.S. Companies in 2014. The Heritage Foundation – National Security and Defense (4289):1–5, URL http://www.heritage.org/research/reports/2014/10/cyber-attacks-on-us-companies-in-2014, issue Brief
59. Wang YM, Roussev R, Verbowski C, Johnson A, Wu MW, Huang Y, Kuo SY (2004) Gatekeeper: Monitoring Auto-Start Extensibility Points (ASEPs) for Spyware Management. In: Proceedings of the 18th USENIX Conference on System Administration, USENIX Association, Berkeley, CA, USA, LISA '04, pp 33–46
60. Weaver N, Paxson V, Staniford S, Cunningham R (2003) A Taxonomy of Computer Worms. In: Proceedings of the 2003 ACM Workshop on Rapid Malcode (WORM), ACM, New York, NY, USA, WORM '03, pp 11–18, DOI 10.1145/948187.948190
61. Yan W, Zhang Z, Ansari N (2008) Revealing Packed Malware. IEEE Security & Privacy 6(5):65–69, DOI 10.1109/MSP.2008.126
62. Yaneza J (2015) GamaPoS: The Andromeda Botnet Connection. Tech. rep., Trend Micro, available at http://documents.trendmicro.com/assets/GamaPOS_TechnicalBrief1.pdf.
63. Zetter K (2010) TJX Hacker Gets 20 Years in Prison. [Online], http://www.wired.com/2010/03/tjx-sentencing/