Threat Spotlight: Holiday Greetings from Pro PoS – Is your payment card data someone else's Christmas present?

By Earl Carter

Published: 2015-12-17 · Archived: 2026-04-05 14:54:53 UTC

Thursday, December 17, 2015 11:23

The post was authored by Ben Baker and Earl Carter.

Payment cards without an EMV chip have reached their end-of-life. Point of Sale (PoS) malware, such as PoSeidon, has continued to threaten businesses. The news is continually filled with stories of payment card data being stolen through a breach in the company's PoS system. From high-end hotels to large retail firms, threat actors are attacking PoS systems in the attempt to capture payment card data. PoS Malware is just another threat category that Talos is monitoring and developing defenses against. In this post, we will examine the functionality of Pro PoS so that you can better understand how this malware can be used to exfiltrate payment card information and potentially other valuable information from your network.

Beginning in October, merchants in the United States were required to use PoS terminals that provide support for chip-enabled cards or otherwise risk liability for fraudulent charges. These new chip-enabled readers help minimize the chance for PoS malware to steal payment card information because the chip on the payment card generates a single use token. This transition, however has been bumpy at best because the cost of new chip-enabled readers has made it difficult to upgrade to the newer readers. Another loophole is that gas stations have a different timeline and are not required to move to chip-based readers until October 2017. These two factors mean that many establishments still rely on card readers that are not chip-enabled and sending payment card data that can be duplicated and reused.

Pro PoS is simple-to-use PoS malware that is available for purchase, enabling multiple threat actors to easily take advantage of this malware to target businesses. The functionality of Pro PoS seems fairly extensive according to recent press releases. These claims include the following:

1. Tor support
2. Rootkit functionalities
3. Mechanisms to avoid antivirus detection
4. Polymorphic engine

In order to analyze the actual capabilities of Pro PoS, Talos collaborated with Flashpoint, a pioneer in threat intelligence from the Deep & Dark Web. Not all of the claims in the press releases seem to be totally accurate given the Pro PoS version 1.1.5b sample that Talos analyzed. For instance we did not identify any significant mechanisms to avoid antivirus detection, other than a trivial packer that seemed to be more for compression than obfuscation. Unless you include tor2web, we did not find support for Tor. We did not find a Polymorphic engine. And finally, we did observe a rootkit being installed but it did not appear to be used by the malware.

Client Software

The Pro PoS client malware uses a modified version of Alina, which had its source code leaked earlier this year. In this version, the malware utilized a simple packer that does not contain any anti-analysis checks. Given the simplicity of the packer and the fact that it even leaves some of the string in the binary unaltered, it is likely that the packer was meant to simply compress the binary, instead of trying to make the examination of the binary more complicated.

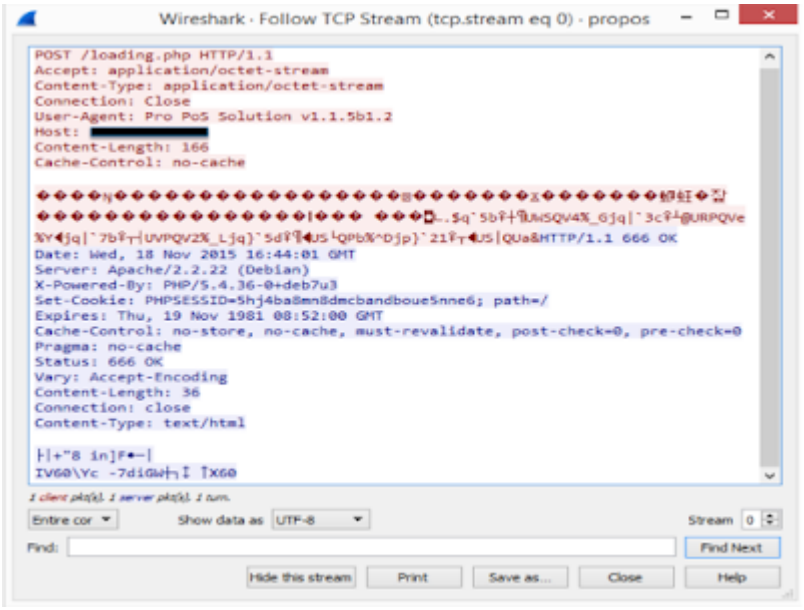
Before execution, Pro PoS copies itself to “%appdata%\914785304\NTProvider.exe”. The folder name is generated by adding the output of rand() to 914785263. The developers failed to initialize the random number generator by calling srand() before calling rand() so the same value will be generated every time that the malware is run. The malware also reads in the original file, RC4 encrypts it using the password “Password”, then drops it to “%appdata%\driver.sys”.

Pro PoS injects explorer.exe with shellcode via WriteProcessMemory. The shellcode attempts to open “%appdata%\ntkrnl”, then RC4 decrypt it, and write it to %appdata%\Installed\windefender.exe. It then uses ShellExecuteA to execute windefender.exe. The Pro PoS variant of the Alina client does not drop a file named ntkrnl, meaning the shellcode will be unable to decrypt and execute a file named windefender.exe.

Pro PoS then iterates through running processes checking against a whitelist, spawning new threads dedicated to memory scraping each process not in the whitelist. This results in several threads running simultaneously, each looking at a different process. The threads use ReadProcessMemory to access the process’s memory to scan through it looking for valid track data like “4012888888881881=18012010123400000000?”. It first looks for the delimiter of “=” or “D” that separates the payment card number from the date. It then verifies the delimiter is preceded by 16 digits, with a valid YYMM date after the delimiter. It checks the 3 digits after the date ensuring the card has service codes 201 or 101, which signify the card can be used internationally with no restrictions. It then uses the Luhn algorithm, which is a checksum used to verify payment card numbers.

Network traffic uses a 1 byte XOR key (0xAA), with version info, software name, botID, action, and PC name. The data after identifying info is XOR'd with an additional 18 byte key derived from other info in the packet.

Sample network traffic:



When the client connects, the controller can send configuration settings like how often to beacon back, and can even respond with a URL, which the client will then use to download a file to the victim's %appdata% path then execute.

Rootkit Functionality

This rootkit with PDB path "c:\drivers\test\objchk_win7_x86\i386\ssdthook.pdb" uses SSDT hooks for the following API:

ZwQueryDirectoryFile - hide files named "windefender.exe"

ZwQuerySystemInformation - hide "windefender.exe" processes

ZwEnumerateValueKey - hide registry keys named "windefender"

SSDT hooks are simple to implement, and effective when hiding on a system. Altering the SSDT on newer versions of Windows (including the 64 bit version of Win7) will trigger patchguard to bugcheck, causing the notorious Blue Screen Of Death (BSOD). Fortunately for the malware developers, PoS systems usually aren't running the latest and greatest versions of Windows. The developers could have used Direct Kernel Object Manipulation (DKOM) which works on newer versions of Windows, but they would still have to bypass driver signing.

This minimalistic rootkit does very little to validate data before using it. It's possible, but unlikely, for pNextEntryOffset to be used before it's given a valid address, meaning 0x00011098 would use the initialized value of Null. Null Pointer Dereference vulnerabilities like this one are often used in exploits for privilege

escalation, though malware with permissions to install drivers do not need to escalate privileges. This bug was almost certainly unintended, and could crash the PoS terminal with the Blue Screen Of Death

```
 .text:00011095    mov     eax, [ebp+pNextEntryOffset]
.text:00011098    mov     ecx, [eax]      ; Possible Null Ptr Dereference
```

Most of the client versions we have examined install the rootkit, but only Joker 1.8 actually uses the filenames and registry key names that are hidden. It's unclear whether the developer forgot to change the rootkit to hide the new filenames, or if he intended to sell the rootkit functionality as a premium feature.

Other Alina Client Variants

We've found several variants of the Alina client, all of which are incredibly similar other than a few string changes. After removing the check for "Pro PoS" in the user-agent, our control panel works for the variants we tested. It wasn't hard to make signatures that detect all of the variants we have, both in ClamAV and Snort. These variants include

1. Joker 1.8
2. Katrina variant of Alina
3. Two unnamed variants

The Joker 1.8 variant is the only version that we analyzed that uses the windefender strings that are hidden by the rootkit. It also incorporates a large number of anti-analysis features including

1. It overwrites the first instruction of "DbgUiRemoteBreakin" and "DbgBreakPoint" with the assembly instruction "retn" which can interfere with debuggers.
2. Uses the intel instruction "vcpext" to detect if it's being run in a VM. Checks to make sure the machine has at least 2 processors. Checks to make sure the system has at least a minimum amount of physical memory.
3. Calls IsDebuggerPresent, CheckRemoteDebuggerPresent, OutputDebugStringA to check for a debugger (Lol this is trivial compared to the other checks).
4. It creates multiple exceptions that if handled by a debugger will fail a check.
5. It uses GetTickCount to check timing for how long it takes to run a code chunk. If it takes too long, it fails the check.
6. It looks for any windows in this list:

- OLLYDBG, VBoxTrayToolWndClass, ID, Tokno_konfig, TDiEfrm, MYDEBUG, 259C91A0, 18467-41, FileMonClass, OWL_Window, HANOLLY, YPOGEiOS, DeFixed, TIdaWindow

1. Any loaded modules in this list:

- Cmdline.dll, BOOKMARK.DLL, pluzina1.dll, pluzina2.dll, pluzina3.dll, pluzina4.dll, procs.dll, realign.dll, 16Edit.DLL, win32_user.plw, win32_stub.plw, linux_stub.plw, wince_stub.plw, mac_stub.plw, DeviareCOM.dll, Nektra.Deviare2.dll, SbieDll.dll, apimonitor-drv-x86.sys.

1. Any running processes in this list:

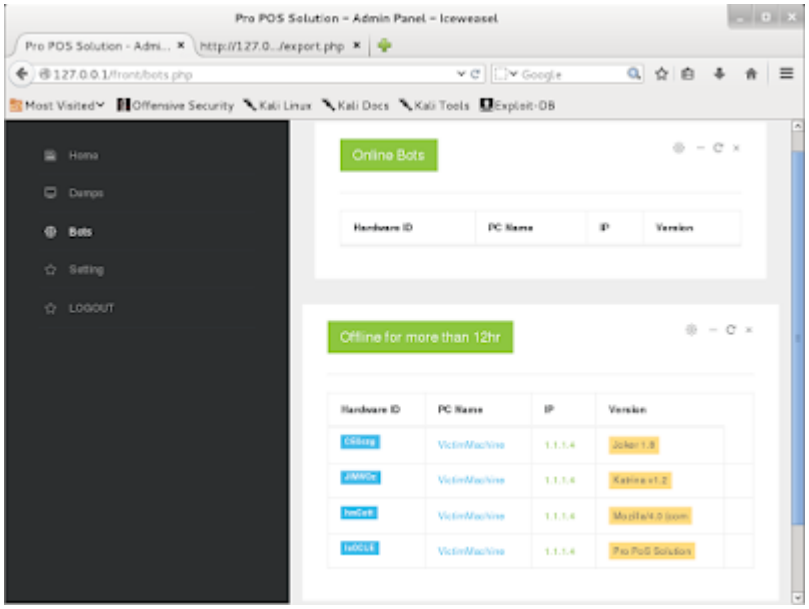
- OLLYDBG.EXE, PEiD.exe, ollydbg.exe, OllyDbg.exe, LordPE.exe, LordPE.exe, ImportREC.exe, CiM's.exe, DeFixed.exe, YGS-DOX.exe, OllyICE.exe, HanOlly_English.exe, HanOlly.exe, HanOlly_Korean.exe, W32DSM89.EXE, WinHex.exe, HIEW32.EXE, XVI32.exe, idag.exe, hiew32.exe, PROCDUMP.exe, FILEMON.EXE, FILEMON.exe, PROCDUMP.EXE, Regmon.exe, ResHacker.exe, exeinfope.exe, eXeScope.exe, DiE.exe, protection_id.exe, EvO_DBG.exe, SbieCtrl.exe, SpyStudio.exe, SbieSvc.exe, apimonitor-x86.exe

1. Any registry keys in this list:
2. HKLM\HARDWARE\ACPI\DSDT\VBOX__
3. HKLM\HARDWARE\ACPI\DSDT\AMIBI
4. It checks for hardware breakpoints by using GetThreadContext on the current thread and checking the contents of Context.Dr0 through Context.Dr3.
5. It checks for software breakpoints at the start of any of these APIs:

- OutputDebugStringA, Process32Next

1. It checks what permissions it has to the process “services.exe”. If it doesn’t like the permissions, it fails the check

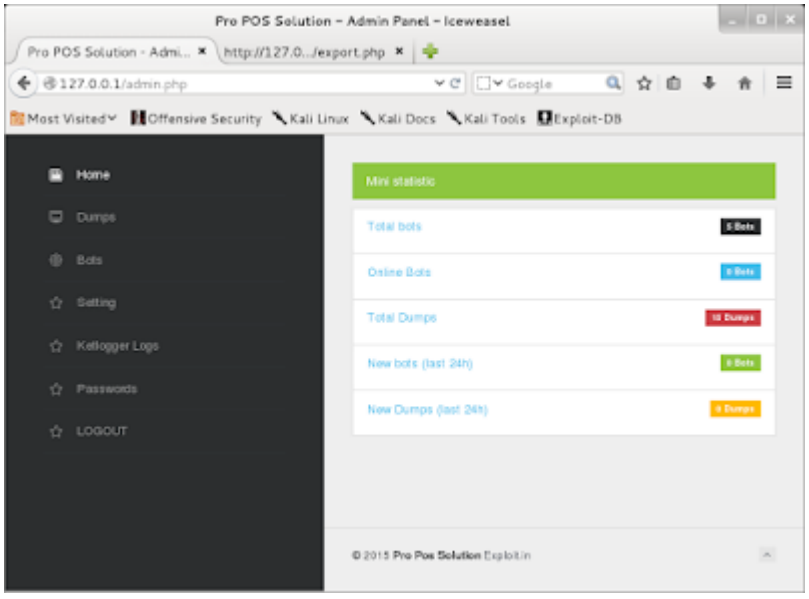
This is a screenshot of the Pro PoS 1.1.5 Control Panel administering Pro PoS and 3 other Alina variants.



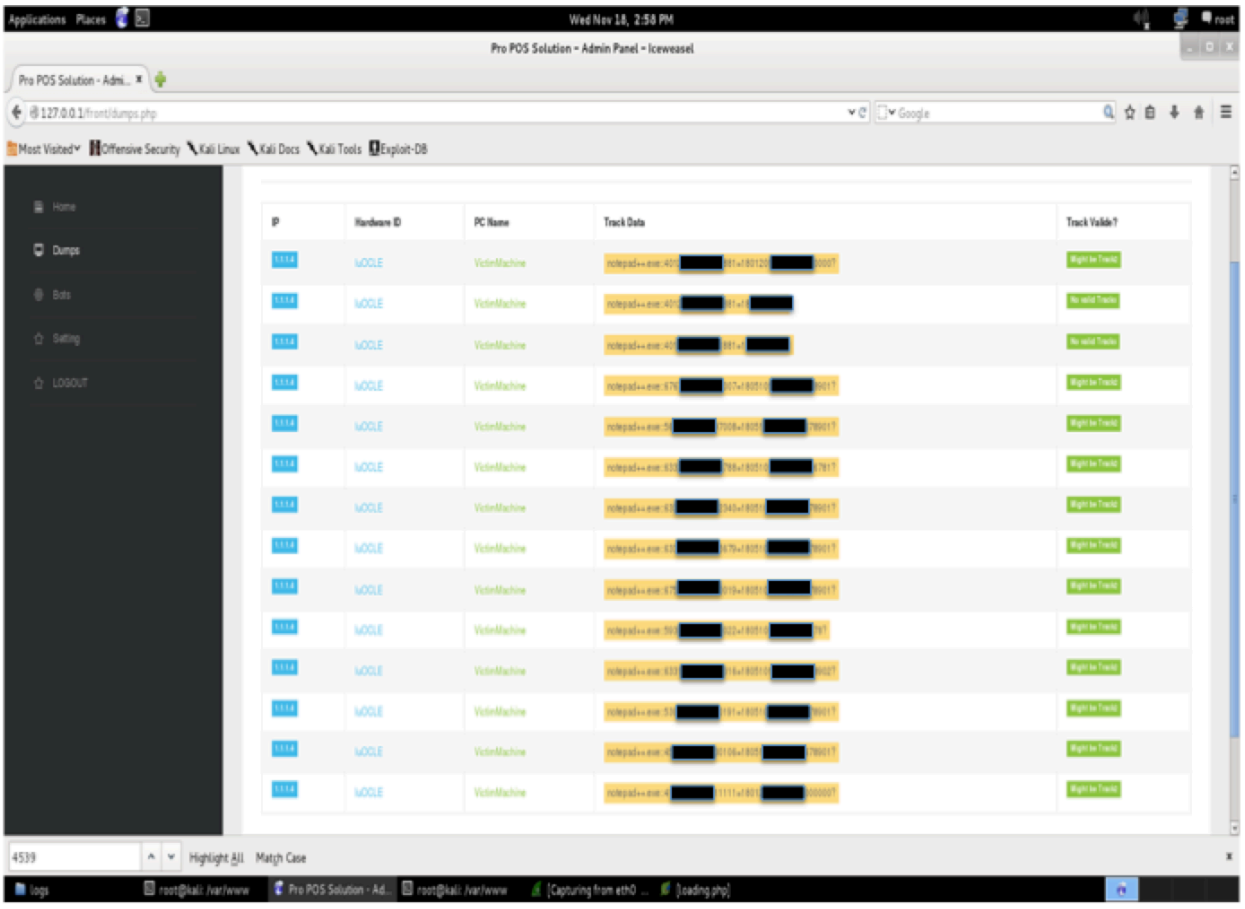
Pro PoS Control Panel

The Control Panel doesn’t use PHP obfuscation, so reversing the network protocol was a breeze. One of the PHP files contains a vulnerability that leads to arbitrary PHP execution. Obviously security wasn’t a major concern when developing this malware. The web page contains links to Passwords and Ketlogger (shown in the

image below), Both of these links simply redirect the user to the home page. These unsupported pages could possibly point to future features that the threat actor plans to add to the control panel.



The Config page allows the controller to force bots to download and execute a file from a URL. This feature appears to be broken and not updating the mysql table that is read from when commands are sent to the bots. Manually adding entries in the mysql table does work. The PHP seems to be the weakest link in this toolkit.



Exfiltrated Data in Control Panel

The login page uses an external image file http[:]//images[.]cooltext[.]com/4465794.png.

Alina Network Client Script

The network client/CNC use a simple binary protocol over standard HTTP for all of it's requests. The first request is that the HTTP User-Agent must begin with 'Pro PoS'. After that, the request's body is then XOR'ed with the static one byte key of 0xAA. The result will be the common request header followed by further XOR "encrypted" data and the key to decrypt it. This lower level is used for the actual payment card track data as well as notifications to download and execute from specific URLs or simple status updates.

The following Ruby shows the header format:

```
 # Client version - WORD
msg =    [0x0102].pack('s')

# Software name  - NULL padded 16 byte string
msg <<   pad("Pro PoS Solution", 16)
```

```
# hardware ID - NULL padded 8 byte string
msg <<   pad("GwWASP", 8)

# 2x bytes but it is unclear what it is used for
msg <<   "EE"

# Action (update, etc) - NULL padded 8 byte string
msg <<   pad("update", 8)

# PC Name - NULL padded 32 byte string
msg <<   pad("WIN7-41424345", 32)

# Total message size - DWORD
msg <<   [data.size + 123].pack('V')

# A simple checksum of the message header
msg <<   checksum(msg)

# Finally, XOR the rest of the data with the hardware ID + 2 bytes + Action
msg <<   xor(data, xor(msg[18..35]), 18)
```

From here, the data sent in the request is stored directly in the database used by the control panel.

Conclusion

Payment cards without EMV chips and businesses who do not yet have chip-enabled PoS terminals have become an unnecessary security risk. As long as PoS terminals rely on payment data stored in the magnetic stripe, threat actors will continue to invest in innovation and development of new malware families to exploit this attack vector. Attackers will continue to target PoS systems and employ various obfuscation techniques in an attempt to avoid detection. Since PoS malware like Pro PoS is available for purchase, it is even easier for threat actors to utilize it to steal payment card data.

Businesses who utilize payment card readers that are not chip-enabled will need to remain extra vigilant and adhere to industry best practices to ensure coverage and protection against these advancing malware threats, especially during the holiday season.

Protecting Users

Snort Rules The following Snort rule will detect Pro PoS. This rule is subject to change pending new information regarding the threat. Please refer to your FireSIGHT Management Center or the Snort

Subscriber Rule Set for the latest rules.

36331

PRODUCT	PROTECTION
AMP	✔
CWS	✔
ESA	✔
Network Security	✔
WSA	✔

Advanced Malware Protection (AMP) is ideally suited to prevent the execution of the malware used by these threat actors.

CWS or WSA web scanning prevents access to malicious websites and detects malware used in these attacks.

The Network Security protection of IPS and NGFW have up-to-date signatures to detect malicious network activity by threat actors.

ESA can block malicious emails sent by threat actors as part of their campaign.
