BlackBerry Cylance vs. Tinba Banking Trojan

blogs.blackberry.com/en/2019/03/blackberry-cylance-vs-tinba-banking-trojan

The BlackBerry Cylance Threat Research Team, Tatsuya Hasegawa

RESEARCH & INTELLIGENCE / 03.13.19 /

Our 'BlackBerry Cylance Versus' series takes an in-depth look at malware from A to Z, from past to present, with a goal of throwing light on how and why threats that may have been active for years still work, and what we, as a security community, can do to combat them.

Tinba (aka. Tina or Zusy), which stands for "Tiny Banker", is a banking Trojan that has targeted Windows computers since 2012. It is a famously small malware, file-wise, with a code base of 20 kB. It is specifically designed to target financial institutions. Tinba steals browsing data, login credentials, and other sensitive information by using Man-in-the-Browser (MitB) attacks.

Tinba performs code injections on running processes to hide itself and achieve persistence. In 2014 the source code for Tinba leaked on an underground cybercrime forum, giving threat actors worldwide access to this powerful malware.

Tinba Analyzed

Our analysis of Tinba was performed on the following hashes:

- E7C0B1BD0DB584D82E3D77F283467C899656075189183B5A8F8431C458E60321
- 0283798A83AA597BF15ED5A59C21E68D66F6789B2ACABBE87DCA9C089608B893
- 83C2B35F72749433E76B16F25A1CA9715B55AA280FB5D158389EAFD17CD0D392

Code Features

The Tinba variants we analyzed have the product/project name "**Dealhoya**". They are compiled with Visual Basic 6 which requires VB6 runtime to run. These variants attempt to camouflage themselves as a flash game. The file description reads *"flash game Lucknow is the capital city of the state of Uttar"* and the filename is *FergusGamez.exe*.

True to its reputation, the Tinba files are tiny, each weighing in at less than 100 kB. The malware files are highly obfuscated as shown in Figure 1:

```
loc_00404EA1: mov var_EC, 00403370h ; "pAFCNfg"
loc_00404EAB: mov var_F4, ebx
loc_00404EB1: call __vbaVarCopy
loc_00404EB3: lea edx, var_F4
loc_00404EB9: lea ecx, var_68
loc_00404EBC: mov var_EC, 00403348h ; "aRrTyweSne"
loc_00404EC6: mov var_F4, ebx
loc_00404ECC: call __vbaVarCopy
loc_00404ECE: lea edx, var_F4
loc_00404ED4: lea ecx, var_50
loc_00404ED7: mov var_EC, 0040339Ch ; "TNrta"
```

Figure 1: Tinba's obfuscated code

Evasion Technique

Tinba can detect and evade virtual environments by calling the following Windows APIs:

- GetDiskFreeSpaceExW
- GlobalMemoryStatusEx
- GetAdaptersAddresses

Tinba also monitors user activity in the active window by calling *GetForegroundWindow*. These functions allow the malware to judge whether the platform is an analysis environment such as sandbox or debugger.

Code Injection

When Tinba executes it creates another process of itself. This second process launches a legit Windows application called **winver.exe** and injects the malicious code into it. Winver.exe is the standard program for displaying Windows version information.

The injected code checks for the presence of *explorer.exe* by finding the window with the *Shell_TrayWnd* class name. If found, the malware attempts to inject secondary code into *explorer.exe*. The secondary code also injects the main Tinba code into all active

processes. When successful, this attack results in ten or more injected process running Tinba in their threads (See Figure 2):

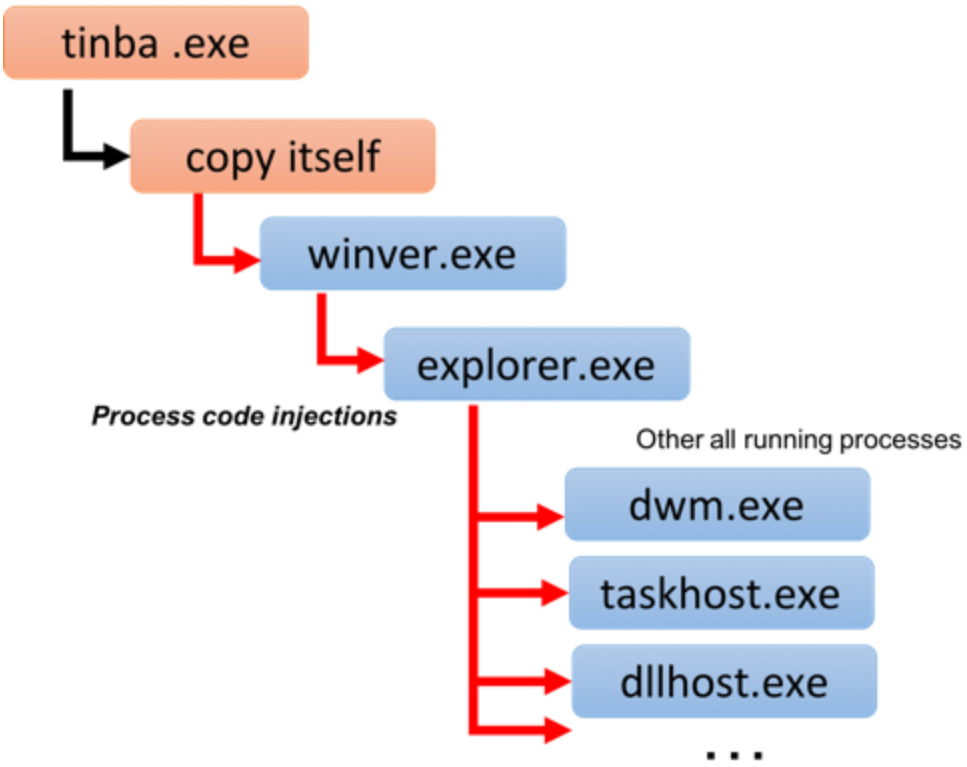

Figure 2: Tinba execution flow

Dropping the EXE File

The infected *explorer.exe* serves as the main process of Tinba. It drops *bin.exe* into %AppData% and adds the following RUN key to achieve persistence:

Registry key: HKCU\Software\Microsoft\Windows\CurrentVersion\Run\(random string)
Registry value: %AppData%\(random string)\bin.exe

The random string is eight alphanumeric characters ([0-9A-Z]{8} in regular expression) that are unique to each infected machine. *Bin.exe* is the polymorphic version of Tinba, which means the file hash differs for each infection. The malware also creates directories using the random string then sets the hidden attribute:

- %AppData%\Local\Packages\windows_ie_ac_001\AC\(random string)
- %AppData%\LocalLow\(random string)
- %AppData%\(random string)

Command and Control (C2) Communication

Tinba connects to the C2 server to post the 157 bytes of encrypted system information using the HTTP POST method:

- C2 URL: hxxp://recdataoneveter[.]cc/vet7sdfh678sdjjs7er0k/
- DNS resolved IP address: 216[.]218[.]185[.]162

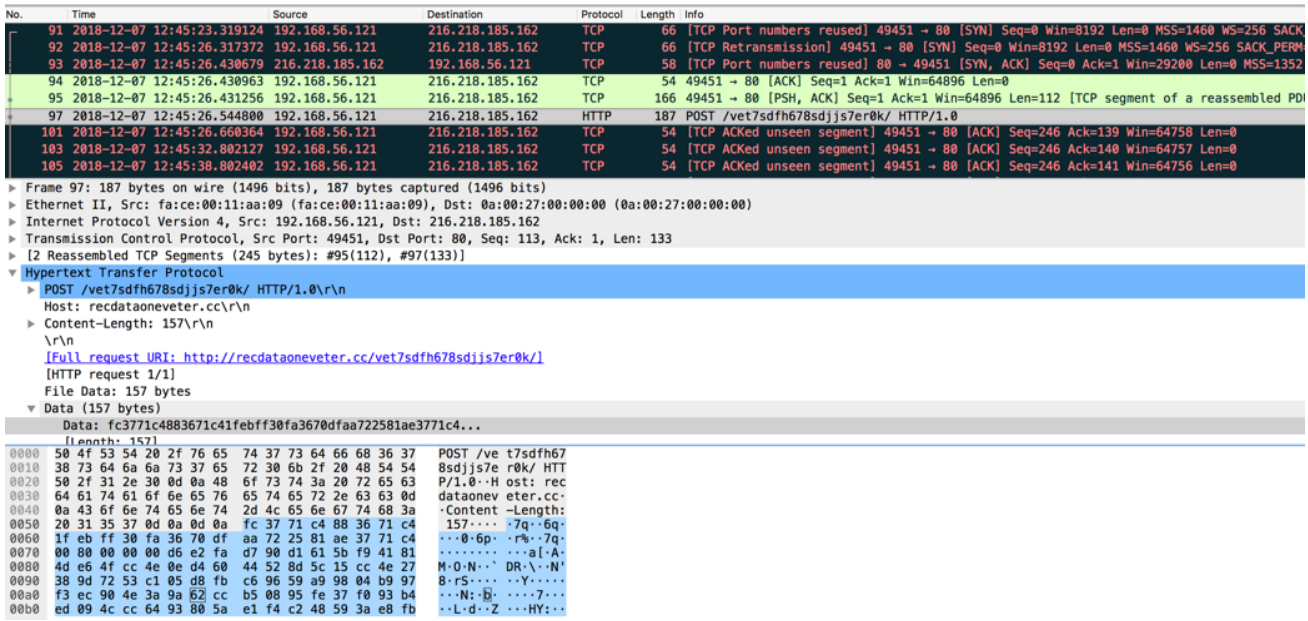

Figure 3: Tinba C2 protocol

Some variants of Tinba use DGA (Domain Generation Algorithm) domains. This process uses a hardcoded domain as the seed to generate short-lived DGA domains which obfuscate C2 communications. Tinba also uses Fast Flux domains where allocated IP address change frequently. In the above case, the server was not available at the time of our investigation because it was taken down.

When the C2 connection succeeds, Tinba downloads additional payloads like a browser injection module and a new banking target URL list. It will also install any pending updates.

MITB for Stealing Bank Account Credentials

The infected *explorer.exe* seeks out Internet Explorer and Firefox so Tinba can use Man-in-the-Browser (MitB) attacks to steal bank account information. Tinba targets accounts related to financial institutions, Google, Facebook, and Microsoft. Malicious code injected into the browser will monitor credential information as it is entered into login pages or steal it from the browser cache. Tinba encrypts stolen data with the RC4 algorithm and sends it to the C2 server.

Indicators of Compromise (IOCs)

- **Sample Hashes**
  E7C0B1BD0DB584D82E3D77F283467C899656075189183B5A8F8431C458E60321
  0283798A83AA597BF15ED5A59C21E68D66F6789B2ACABBE87DCA9C089608B893
  83C2B35F72749433E76B16F25A1CA9715B55AA280FB5D158389EAFD17CD0D392

- **Filenames**
  - FergusGamez.exe

- **C2s/IPs**
  - hxxp://recdataoneveter[.]cc/vet7sdfh678sdjjs7er0k/
  - 216[.]218[.]185[.]162

- **Registry**
  - key: HKCU\Software\Microsoft\Windows\CurrentVersion\Run\[0-9A-Z]{8}§
    - value: %AppData%\[0-9A-Z]{8}\bin.exe

- **Create Folder**
  - %AppData%\Local\Packages\windows_ie_ac_001\AC\[0-9A-Z]{8}
  - %AppData%\LocalLow\[0-9A-Z]{8}
  - %AppData%\[0-9A-Z]{8}

BlackBerry Cylance Stops Tinba

Blackberry Cylance offers a predictive advantage over zero-day threats and is also effective against legacy malware like Tinba. Blackberry Cylance uses artificial intelligence (AI) agents trained for threat detection on millions of both safe and unsafe files. This allows BlackBerry Cylance prevent Tinba from executing based on the analysis of several malicious file attributes instead of a specific file signature.

About The BlackBerry Cylance Threat Research Team

The BlackBerry Cylance Threat Research team examines malware and suspected malware to better identify its abilities, function and attack vectors. Threat Research is on the frontline of information security and often deeply examines malicious software, which puts us in a unique position to discuss never-seen-before threats.

About Tatsuya Hasegawa

Senior Threat Researcher at BlackBerry Cylance

Tatsuya Hasegawa is a Senior Threat Researcher in APAC at BlackBerry, and is responsible for malware analysis and sandbox technology. He has practical experience in the both managed security service provider as a security analyst and CSIRT as an incident handler. His certifications include: GREM, GCIH, GCFA, GXPN, GPEN and CISSP.

Back