Vadokrist: A wolf in sheep's clothing

By ESET Research

Vadokrist is a Latin American banking trojan that ESET has been tracking since 2018 and that is active almost exclusively in Brazil. In this installment of our series, we examine its main features and some connections to other Latin American banking trojan families.

Vadokrist shares several important features with families we have described earlier in the series, namely Amavaldo, Casbaneiro, Grandoreiro and Mekotio. We recently published a white paper dedicated to documenting the similarities between Latin American banking trojans, whereas this blogpost series focuses more on the detailed analysis of one family at a time.

Characteristics

Vadokrist is written in Delphi. One of the most notable characteristics is the unusually large amount of unused code in the binaries. After further examination, we believe this is an attempt to evade detection and dissuade or slow analysis. We were able to link some of the code to existing Delphi projects, such as QuickReport.

Vadokrist stores strings inside string tables. It used to contain an implementation of a string table identical to Casbaneiro (illustrated in Figure 1); however, some recent versions of this banking trojan switched to using multiple string tables, each for a different purpose (list of targets, general configuration, backdoor command names, etc.).

```
CreateStringTable proc near
push    ebx
push    esi
mov     ebx, off_93B264
mov     dl, 1
mov     eax, ds:VMT_497960_TStringList ; TStringList_Self
call    TStringList_Create
mov     esi, eax
mov     eax, [ebx]
mov     [eax+410h], esi
mov     eax, esi        ; TStringList_Self
mov     edx, offset a187fc7eb167f ; "187FC7EB167F"
mov     ecx, [eax]
call    [ecx+TStringList.TStringList_Add_0x3c]
mov     eax, [ebx]
mov     eax, [eax+410h] ; TStringList_Self
mov     edx, offset a1802 ; "1802"
mov     ecx, [eax]
call    [ecx+TStringList.TStringList_Add_0x3c]
mov     eax, [ebx]
mov     eax, [eax+410h] ; TStringList_Self
mov     edx, offset a187fc7eb15e1 ; "187FC7EB15E1"
mov     ecx, [eax]
call    [ecx+TStringList.TStringList_Add_0x3c]
●
●
●
mov     eax, [ebx]
mov     eax, [eax+40Ch] ; TStringList_Self
mov     edx, offset a6726e84fad5c14 ; "6726E84FAD5C14"
mov     ecx, [eax]
call    [ecx+TStringList.TStringList_Add_0x3c]
pop     esi
pop     ebx
retn
CreateStringTable endp
```

Figure 1. String table implementation in earlier Vadokrist binaries

The vast majority of Latin American banking trojans collect information about the victim machine (typically computer name and version of the Windows OS) when first run. The only information Vadokrist collects is the victim's username and it does so only after initiating an attack on a targeted financial institution, not, unlike most other Latin American banking trojans, at install time.

To ensure persistence, Vadokrist utilizes either a Run key or it creates a LNK file in the startup folder.

Its backdoor capabilities are typical for this type of threat, being able to manipulate the mouse and simulate keyboard input, log keystrokes, take screenshots, and restart the machine. It is also able to prevent access to some websites, which it does in a rather clumsy way by killing the browser process when the victim attempts to visit such websites. We believe this technique is used to prevent the victims from accessing their online bank accounts once the attackers have compromised it, aiding them in retaining control.

Cryptography

The majority of Vadokrist binaries implement a cryptographic algorithm we have seen in other Latin American banking trojans (namely Amavaldo and Casbaneiro) that we have dubbed TripleKey. Vadokrist uses this algorithm to protect its strings, and occasionally payloads and remote configurations as well (we dig deeper into this topic later). For clarity, we have implemented the algorithm in Python, as seen in Figure 2.

```
 def decrypt_payload(data_enc, key1, key2, key3):
    data_dec = str()

    for c in data_enc:
                x = data_enc[i] ^ (key3 >> 8) & 0xFF
                data_dec += chr(x)
                key3 = ((x + key3) & 0xFF) * key1 + key2

    return data_dec

def decrypt_string(data_enc, key1, key2, key3):
    data_dec = str()

    for c in data_enc:
                x = data_enc[i] ^ (key3 >> 8) & 0xFF
                data_dec += chr(x)
                key3 = ((data_enc[i] + key3) & 0xFFFF) * key1 + key2

    return data_dec
```

Figure 2. TripleKey encryption scheme used by Vadokrist to protect strings, payloads and remote configurations

Besides that, we have seen Vadokrist using RC4 in some of its recent binaries and, in the past, TwoFish as well. This is quite rare among Latin American banking trojans as most of them never use generally known cryptographic algorithms.

Distribution

MSI overload

Recent spam emails distributing Vadokrist contain two nested ZIP archives that contain two files – an MSI installer and a CAB archive. If a victim executes the MSI installer, it locates the CAB archive and extracts its

contents (an MSI loader) to disk. It then executes an embedded JavaScript file that adds a Run key entry, making sure the MSI loader is executed on system startup. Finally, the script restarts the machine. On startup, the MSI loader executes an embedded DLL – the Vadokrist banking trojan. The whole process is illustrated in Figure 3. Notice that no actual downloader is in place; the banking trojan is distributed directly by these spam emails.

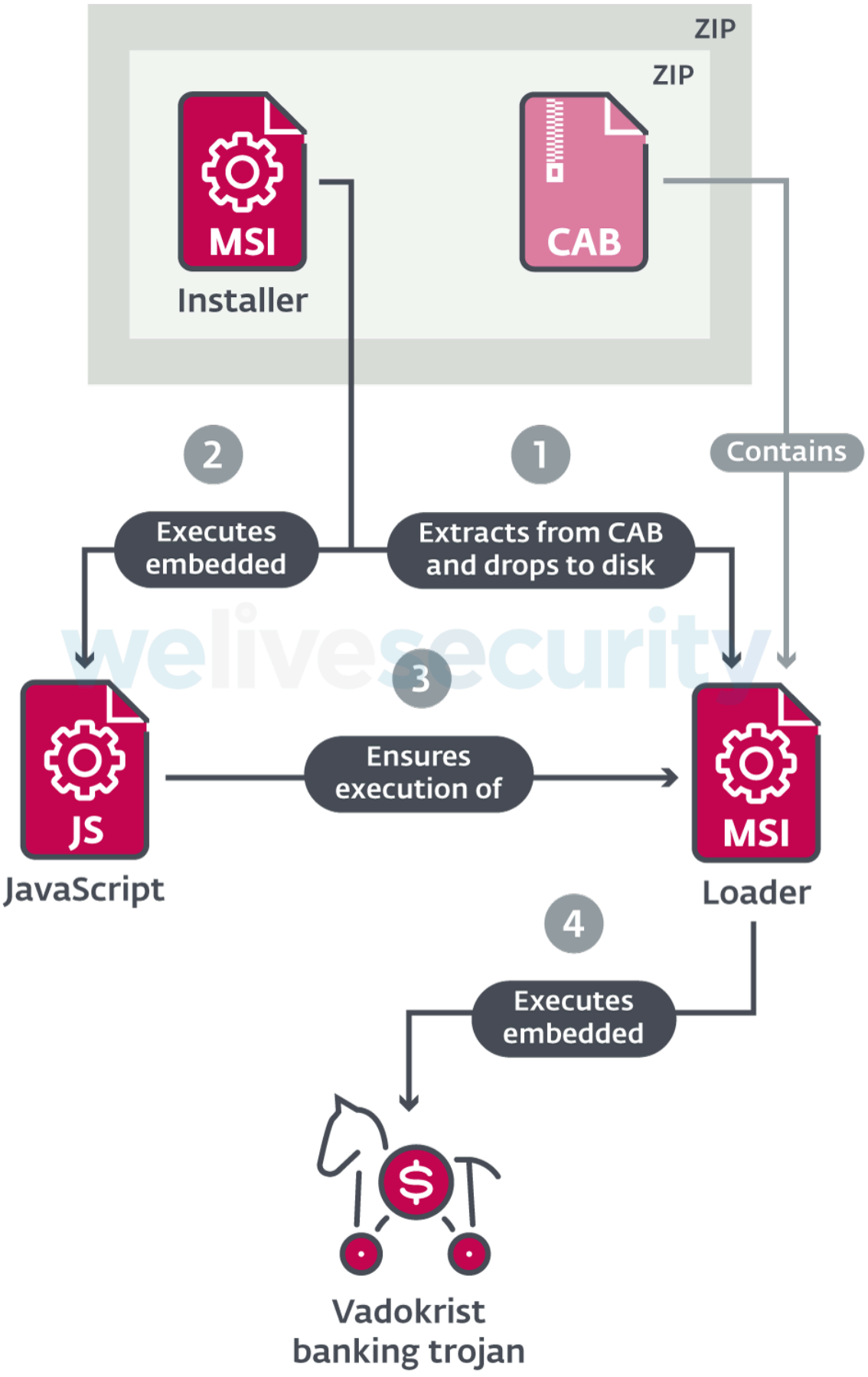
ZIP
ZIP
MSI
Installer
CAB
Contains
2
1
Executes embedded
Extracts from CAB and drops to disk
3
Ensures execution of
JS
JavaScript
MSI
Loader
4
Executes embedded
Vadokrist banking trojan

Figure 3. Execution chain recently used by Vadokrist

The JavaScript file is worth mentioning because of its obfuscation. It leverages how the comma operator (,) works in JavaScript and abuses it to greatly reduce readability and possibly to bypass emulation. It obfuscates conditions using the logical AND operator (&&) in a similar manner. See an example in Figure 4.

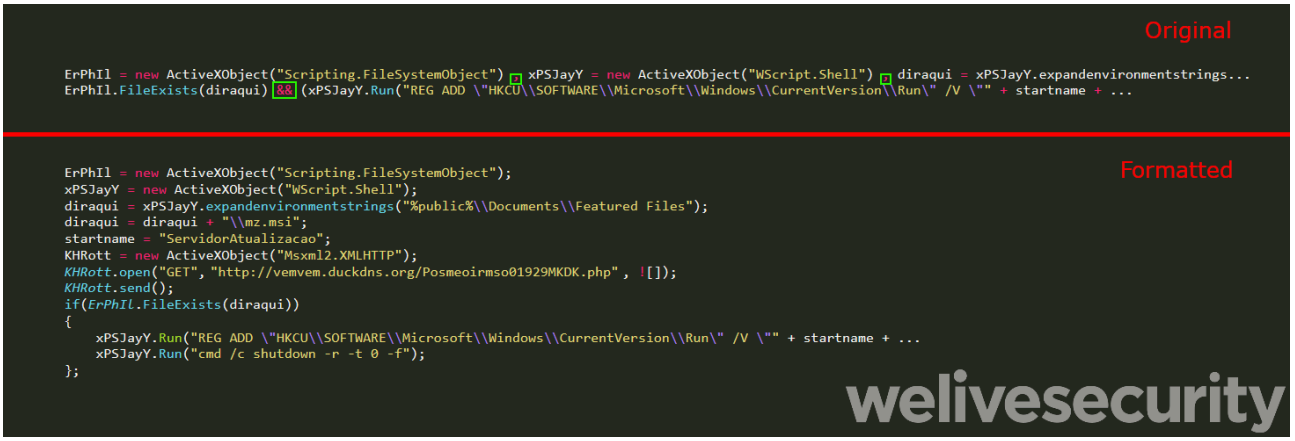

Figure 4. JavaScript installer used by Vadokrist. The bottom part shows the script with transformed operators for better readability.

Older distribution & execution techniques

We observed Vadokrist, like most other Latin American banking trojans, using several implementations of the typical distribution chain. We won't cover all of them, but two are worth mentioning. We have seen Vadokrist sharing a Delphi downloader with Grandoreiro and a whole distribution chain with Mekotio – in fact, the one marked as Chain 1 in our blogpost about Mekotio.

Vadokrist occasionally relies on DLL side-loading with a specific injector to decrypt and execute the banking trojan. This injector is identical to the one used by Amavaldo and implements the aforementioned TripleKey algorithm for data decryption.

Remote configuration

Vadokrist utilizes remote configuration both in downloaders and the actual banking trojan, usually hosted on public storage services such as GitHub.

The configuration file is usually encrypted, either by TripleKey or RC4. In Figure 5, you can see that in both cases the data can be decrypted without any additional knowledge – in the case of the TripleKey method, we can extract all three keys from the end of the string and in the case of RC4, we can derive the key from the password. In the case of the latter, the encrypted data is further encoded by base64.

The delimiter also changes from time to time. So far, we have seen three different characters used: "|", "!" and "/".

Figure 5. Encrypted remote configuration files used by Vadokrist

Now that we know how to decrypt the configuration file, let's examine its contents. In the case of the banking trojan, the result is easy to understand, as it is the IP address of a C&C server. For downloaders, the format is a bit more complex, as illustrated in Figure 6.

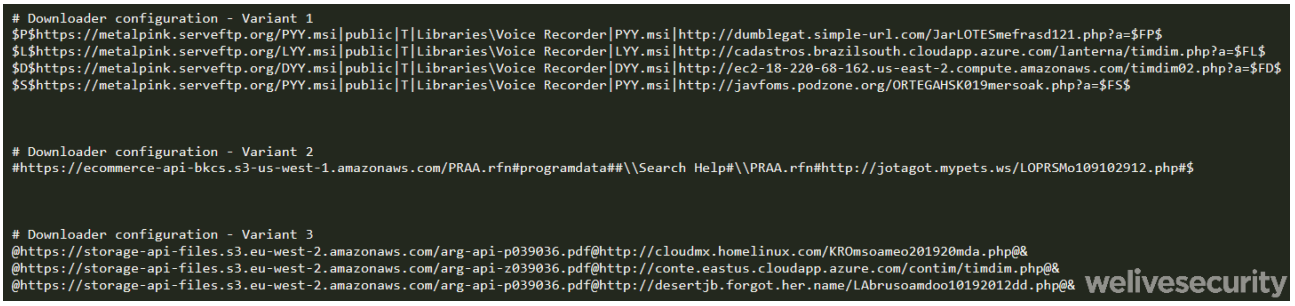

Figure 6. Different formats of remote configuration used by Vadokrist downloaders

To simplify, we recognize a configuration with an optional ID and six fields:

- [mandatory] The URL to download the banking trojan from
- [optional] Special folder (first part of the installation path)
- [optional] Installation flag (described below)
- [optional] Path (second part of the installation path)
- [optional] Filename (third and final part of the installation path)
- [mandatory] Notification URL

Two of these fields may require further explanation. If the installation flag is set to "T", all three parts of the installation path will be used; otherwise the first one will be ignored. The only thing that is sent to the *notification* URL is whether an application Core.exe is running – a check familiar from other Latin American banking trojans that try to detect the presence of anti-fraud software Warsaw GAS Tecnologia.

You can see that the first variant uses all the fields, the second one does not use the ID, and the third one only uses the two mandatory URL fields. The delimiter usage here is a bit more complex, as one delimiter is used to separate different entries and a different one to separate fields of an entry. Additionally, you can see that the delimiters change here as well.

The dynamically changing format of the configuration file indicates Vadokrist is under active and continuous development.

Conclusion

In this blogpost, we have dissected Vadokrist, a Latin American banking trojan that is focused on Brazil. We have shown that it has typical characteristics of a Latin American banking trojan – it is written in Delphi, offers backdoor functionality and targets financial institutions. Its main deviation from the typical implementation is that it does not collect information about victims right after successfully compromising their machines.

We have covered its encryption schemes, distribution and execution methods and remote configuration formats. Vadokrist seems to be connected to Amavaldo, Casbaneiro, Grandoreiro and Mekotio, other Latin American banking trojans described earlier in our series.

For any inquiries, contact us at threatintel@eset.com. Indicators of Compromise can also be found in our GitHub repository.

Indicators of Compromise (IoCs)

Hashes

Campaign “MSI overload”

SHA-1	Description	ESET detection name
D8C6DDACC42645DF0F760489C5A4C3AA686998A1	MSI installer	JS/TrojanDownloader.Banload.ABD
01ECACF490F303891118893242F5600EF9154184	MSI loader	Win32/Spy.Vadokrist.T
F81A58C11AF26BDAFAC1EB2DD1D468C5A80F8F28	Vadokrist banking trojan	Win32/Spy.Vadokrist.T

Other

SHA-1	Description	ESET detection name
8D7E133530E4CCECE9CD4FD8C544E0913D26FE4B	Vadokrist banking trojan	Win32/Spy.Vadokrist.AF
AD4289E61642A4A724C9F44356540DF76A35B741	Vadokrist banking trojan	Win32/Spy.Vadokrist.T

SHA-1	Description	ESET detection name
BD71A9D09F7E445BE5ACDF412657C8CFCE0F717D	Vadokrist banking trojan	Win32/Spy.Vadokrist.AD
06C0A039DEDBEF4B9013F8A35AACD7F33CD47524	Downloader (MSI/JS)	JS/TrojanDownloader.Banload.AAO
FADA4C27B78DDE798F1E917F82226B983C5B74D8	Downloader (Delphi)	Win32/Spy.Vadokrist.Y
525FCAA13E3867B58E442B4B1B612664AFB5A5C0	Injector shared with Amavaldo	Win32/Spy.Amavaldo.L

Recent C&C servers

- 104.41.26[.]216
- 104.41.41[.]216
- 104.41.47[.]53
- 191.232.212[.]242
- 191.232.243[.]100
- 191.235.78[.]249
- 191.237.255[.]155
- 191.239.244[.]141
- 191.239.245[.]87
- 191.239.255[.]102

MITRE ATT&CK techniques

Note: This table was built using version 8 of the MITRE ATT&CK framework.

Tactic	ID	Name	Description
Resource Development	T1583.001	Acquire Infrastructure: Domains	Vadokrist registers its own domains to be used as C&C servers.
	T1587.001	Develop Capabilities: Malware	Vadokrist is operated by the same group that develops it.
Initial Access	T1566.001	Phishing: Spearphishing Attachment	Vadokrist is distributed as a spam attachment.
Execution	T1059.001	Command and Scripting Interpreter: PowerShell	Vadokrist uses PowerShell in some distribution chains.

Tactic	ID	Name	Description
	T1059.005	Command and Scripting Interpreter: Visual Basic	Vadokrist uses VBScript in some distribution chains.
	T1059.007	Command and Scripting Interpreter: JavaScript/JScript	Vadokrist uses JavaScript in its recent distribution chains.
	T1204.002	User Execution: Malicious File	Vadokrist relies on the user to execute the malicious binary.
Persistence	T1547.001	Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder	Vadokrist ensures persistence via Run key or LNK file in the startup folder.
Defense Evasion	T1140	Deobfuscate/Decode Files or Information	Vadokrist is often distributed encrypted and encrypts its remote configuration.
	T1574.002	Hijack Execution Flow: DLL Side-Loading	Vadokrist is sometimes executed by this technique.
	T1036.005	Masquerading: Match Legitimate Name or Location	Vadokrist masquerades as legitimate software.
	T1218.007	Signed Binary Proxy Execution: Msiexec	Vadokrist uses the MSI format for execution.
Credential Access	T1056.001	Input Capture: Keylogging	Vadokrist can capture keystrokes.
Discovery	T1010	Application Window Discovery	Vadokrist looks for bank-related windows based on their names.
	T1057	Process Discovery	Vadokrist tries to discover anti-fraud software by process name.
	T1082	System Information Discovery	Vadokrist discovers victim’s username.
	T1113	Screen Capture	Vadokrist can take screenshots.
Command and Control	T1132.002	Data Encoding: Non-Standard Encoding	Vadokrist communicates via a custom protocol encrypted with the TripleKey algorithm.
Exfiltration	T1041	Exfiltration Over C2 Channel	Vadokrist exfiltrates data via C&C server.
