Threat Assessment: Royal Ransomware

unit42.paloaltonetworks.com/royal-ransomware/

Doel Santos, Daniel Bunce, Anthony Galiette May 9, 2023

By Doel Santos, Daniel Bunce and Anthony Galiette

May 9, 2023 at 6:00 AM

Category: Ransomware, Threat Briefs and Assessments

Tags: Cortex XDR, next-generation firewall, Royal Ransomware, WildFire

This post is also available in: 日本語 (Japanese)

Executive Summary

Royal ransomware has been involved in high-profile attacks against critical infrastructure, especially healthcare, since it was first observed in September 2022. Bucking the popular trend of hiring affiliates to promote their threat as a service, Royal ransomware operates as a private group made up of former members of Conti.

The Unit 42 team has observed this group compromising victims through a BATLOADER infection, which threat actors usually spread through search engine optimization (SEO) poisoning. This infection involves dropping a Cobalt Strike Beacon as a precursor to the ransomware execution. Unit 42 incident responders have participated in 15 cases involving Royal ransomware in the last 9 months.

Royal ransomware also expanded their arsenal by developing an ELF variant to impact Linux and ESXi environments. The ELF variant is quite similar to the Windows variant, and the sample does not contain any obfuscation. All strings, including the RSA public key and ransom note, are stored as plaintext.

Palo Alto Networks customers receive protections against ransomware used by the Royal ransomware group from Cortex XDR, as well as from the WildFire Cloud-Delivered Security Service for the Next-Generation Firewall. The Unit 42 Incident Response team can also be engaged to help with a compromise or to provide a proactive assessment to lower your risk.

Related Unit 42 Topics **Ransomware**, **Malware**

Table of Contents

Overview

The Royal ransomware group was first observed in September 2022, compromising victims and using multi-extortion to pressure victims to pay their fee. Before their first appearance, this group had been linked to a previous ransomware family named Zeon, starting in January of the same year.

Unlike major ransomware groups like LockBit 3.0, which typically operate as a ransomware-as-a-service (RaaS) by hiring affiliates and promoting their RaaS model, we have not observed this particular group using a similar approach. It is suspected that this group is made up mainly of former members of the Conti ransomware group, who operate covertly and behind closed doors. The ex-members that formed this group are known as Team One.

Because some of the people behind this threat were part of the development of Ryuk (discovered in 2018), which is the predecessor of Conti, they have many years of experience. This means they have a solid base for carrying out attacks and know what works when extorting victims. Perhaps due to this experience, the group has already impacted numerous organizations across the globe. We've observed them making demands up to $25 million dollars in BTC.

Royal also has frequently threatened certain critical infrastructure sectors, such as manufacturing and healthcare. In a few months in 2022, the group impacted 14 manufacturing organizations, according to their leak site, and then followed up by publicizing claims of attacking 26 additional manufacturing organizations in 2023. They have also impacted eight healthcare organizations since their inception. The U.S. Department of Health and Human Services issued a warning about the threat Royal ransomware poses to the healthcare sector in January 2023.

Royal also has been one of the ransomware groups disrupting the education industry. We observed that they impacted 14 organizations in the education sector, including school districts and universities. In fact, in just the first few days of May 2023, the group had already impacted four educational institutions.

This group has leveraged their leak site to publicly extort victims into paying the ransom, as shown in Figure 1. The Royal group will harass victims until the payment is secured, using techniques such as emailing victims and mass-printing ransom notes.

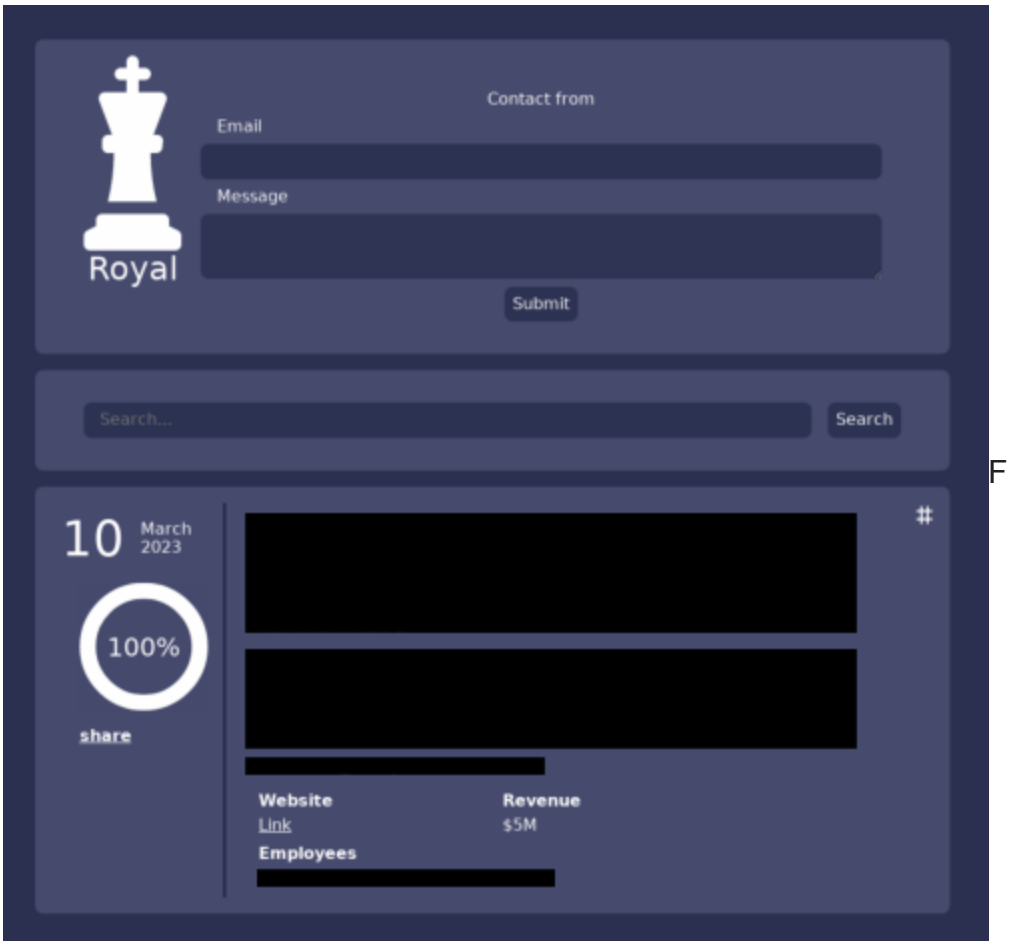


Figure 1. Royal ransomware leak site.

The Royal ransomware threat actor has an active Twitter account that was created in October 2022, called “LockerRoyal.” Most of the account content is announcements of compromised victims, tagging the victim’s Twitter account. In some cases, the threat actor will also reply to those same announcements.

It’s not unusual to see threat actor groups create social media accounts to keep spreading their brand and announcements. It’s clear that this group is trying to get attention from multiple organizations through any means necessary.

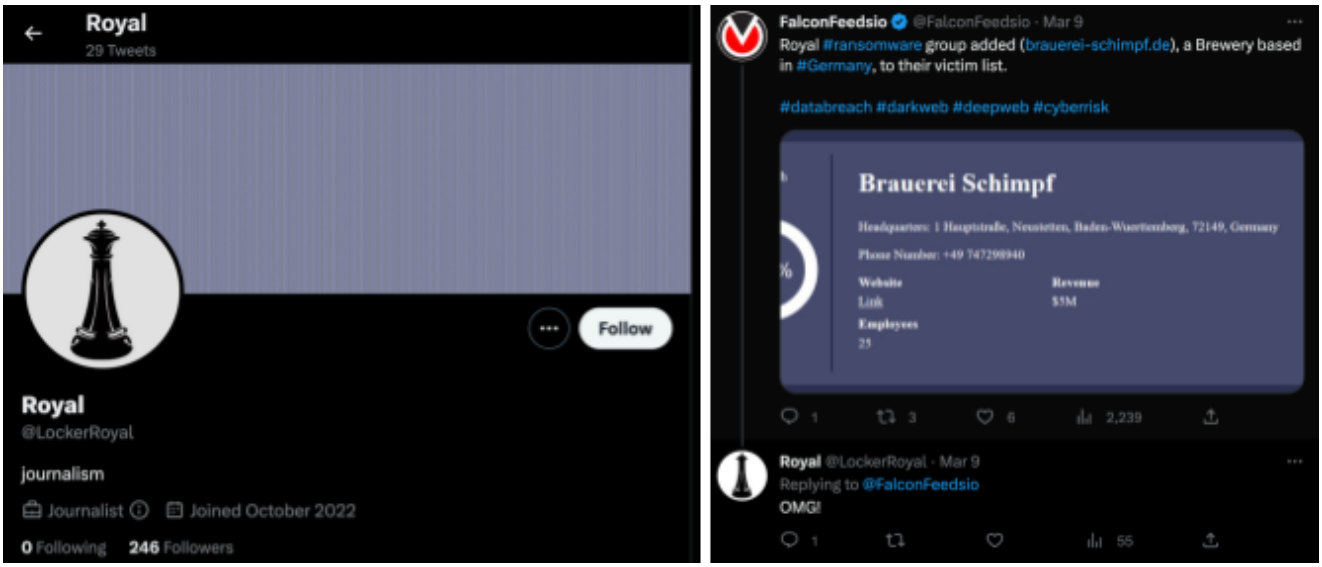


Figure 2. Royal Twitter account replying to an alert.

This particular ransomware group has been observed using multiple initial access vectors to secure access into vulnerable systems, such as the following:

- Callback phishing
- SEO poisoning
- Exposed remote desktop protocol (RDP) accounts
- Compromised credentials

Once access is secured, this group uses multiple tools to support the intrusion operation, like the TCP/UDP tunnel Chisel and the Active Directory query tool AdFind, among others.

Victimology

Royal ransomware has impacted a variety of industries, including small businesses and large corporations alike. Based on information from their leak site and public reporting outlets, we can see that Royal ransomware has impacted industries such as manufacturing, as well as wholesale and retail. Since 2022, Royal ransomware has claimed responsibility for impacting 157 organizations on their leak site.

It's important to note that the impact of Royal ransomware extends beyond just financial losses. There have been instances where the group has targeted critical infrastructure, such as healthcare organizations and agricultural facilities. Since 2022, we have observed this group impacting seven local government entities – like the recent attack on the city of Dallas – in the United States and Europe.

Unit 42 incident responders have participated in 15 cases involving Royal ransomware.

This demonstrates the potential for broader and more severe consequences, as shown in Figure 3.

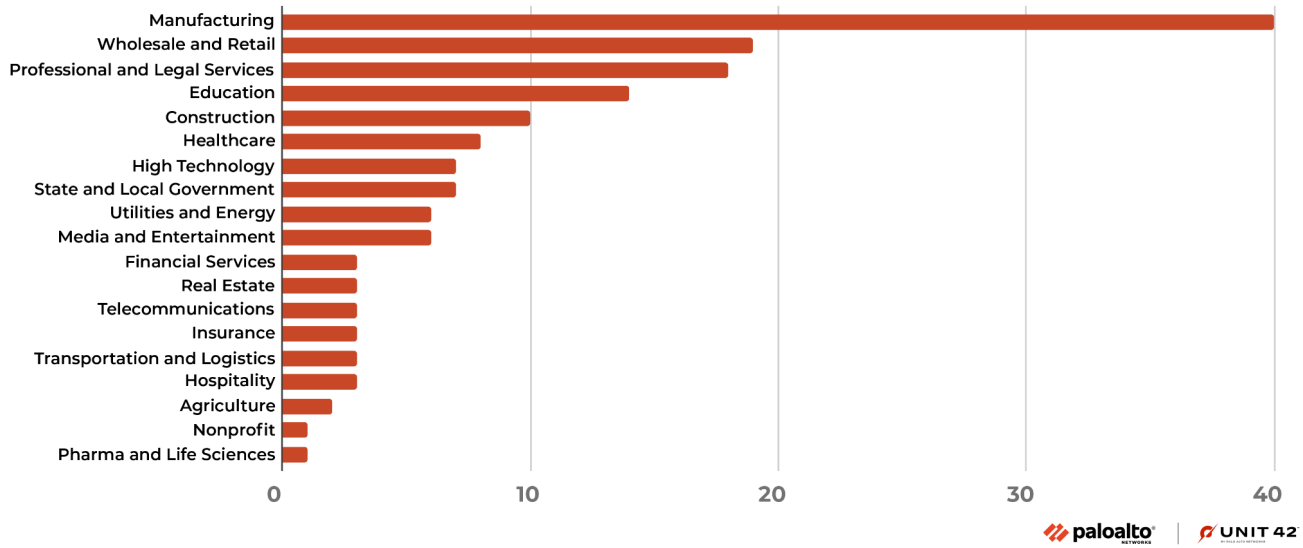


Figure 3. Industries distribution per Royal leak site post.

Most of the organizations impacted by this ransomware are located in the United States, comprising 64% of the impacted organizations. Canada is the country second most impacted by this ransomware family, making the total for North America 73.2%. The next most impacted countries include Germany, the United Kingdom, Brazil, Italy and others (shown in Figure 4).

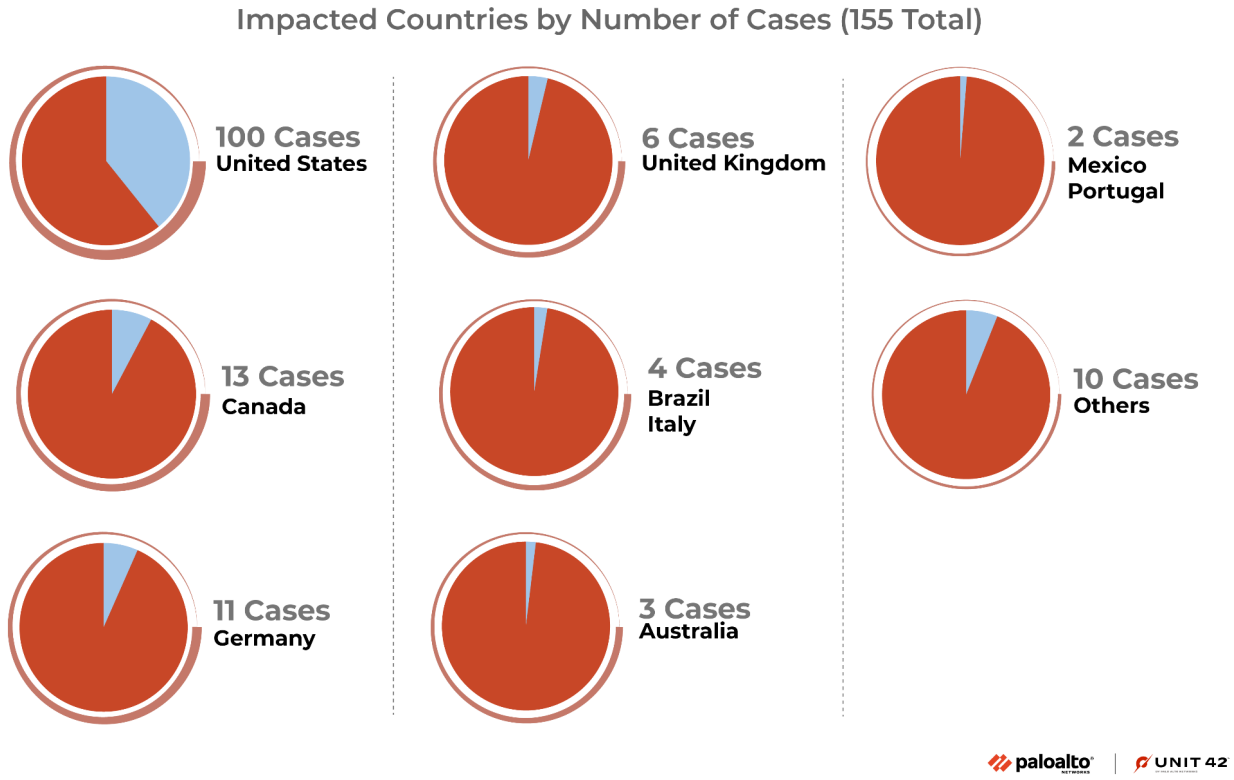


Figure 4. Countries impacted distribution per Royal leak site post.

Infection Chain

There are several different infection chains that lead to Royal ransomware. In some cases, we have observed instances where SEO poisoning and malvertising were used as initial access vectors. The goal of these two methods is to trick a victim into downloading and executing a malicious file that masquerades as legitimate software.

This kicks off a complex infection chain with multiple stages, including PowerShell scripts and MSI files. In certain cases, this leads to infection with BATLOADER.

BATLOADER will then attempt to download further payloads to the infected machine, such as VidarStealer, Ursnif/ISFB and Redline Stealer, as well as legitimate tooling such as the system management tool NSudo and the Syncro remote monitoring and management (RMM) tool. Most importantly, BATLOADER has been seen loading Cobalt Strike, which is often a precursor to ransomware distribution.

In one particular case where we saw Royal ransomware deployed, a snippet of C# code was identified that was originally pulled down from Pastebin. This code, when compiled, would decrypt and load shellcode. The shellcode appeared to be a simple Meterpreter stager that would reach out to an IP address and execute the final Meterpreter beacon. The IP address also hosted a Cobalt Strike server, from which we were able to retrieve the Cobalt Strike configuration.

The configuration contained fairly standard values, although the watermark appeared to be somewhat unique and not randomly generated: 12345. Querying for live Cobalt Strike servers on Shodan with the same watermark returned just over 50 results.

While this unique watermark could indicate a cracked version of Cobalt Strike, examining the domain names for these C2s revealed commonalities across the board. Almost every domain was named to resemble a security company.

For example, the following servers hosted Cobalt Strike beacons with the watermark 12345:

- altocloudzone[.]live
- cloudmane[.]online
- palaltocloud[.]online
- kasperslkyupdate[.]com
- palalto[.]live
- altocdn[.]online
- paloaltokey[.]store
- kasperskyupdates[.]com
- Rapidfinact[.]com

Note that the names above are the work of a threat actor attempting to impersonate legitimate organizations and do not represent actual affiliations with that organization. The threat actor’s impersonation does not imply a vulnerability in the legitimate organizations’

products or services.

Defense Evasion

Unit 42 researchers observed Royal ransomware operators using PowerTool. This is a piece of software that has access to the kernel and is ideal for removing endpoint security software. They also executed batch scripts to disable security-related services, and deleted shadow file copies and logs after successful exfiltration.

Lateral Movement

Unit 42 researchers observed Royal threat actors using the network discovery software NetScan to identify and map out various connected computer resources such as other user targets and shares. In addition to using NetScan, we also observed them using PsExec for conducting lateral movement within the infected environments.

Command and Control (C2)

Unit 42 researchers observed threat actors using various popular legitimate remote management software also used heavily by other ransomware operations to maintain access to the infected environment. The use of Cobalt Strike and related beacons were also observed for C2. An interesting observation of a tool used for maintaining access was the use of Chisel, a TCP/UDP tunneling tool written in Golang.

Exfiltration

Unit 42 researchers observed Royal threat actors using Rclone, a legitimate tool to manage files between two systems, for exfiltrating stolen data before the deployment of ransomware. We found Rclone deployed in folders such as ProgramData, or renamed and masquerading in other folders. One popular filename used was svchost.exe.

Ransomware Functionality

Windows Variant

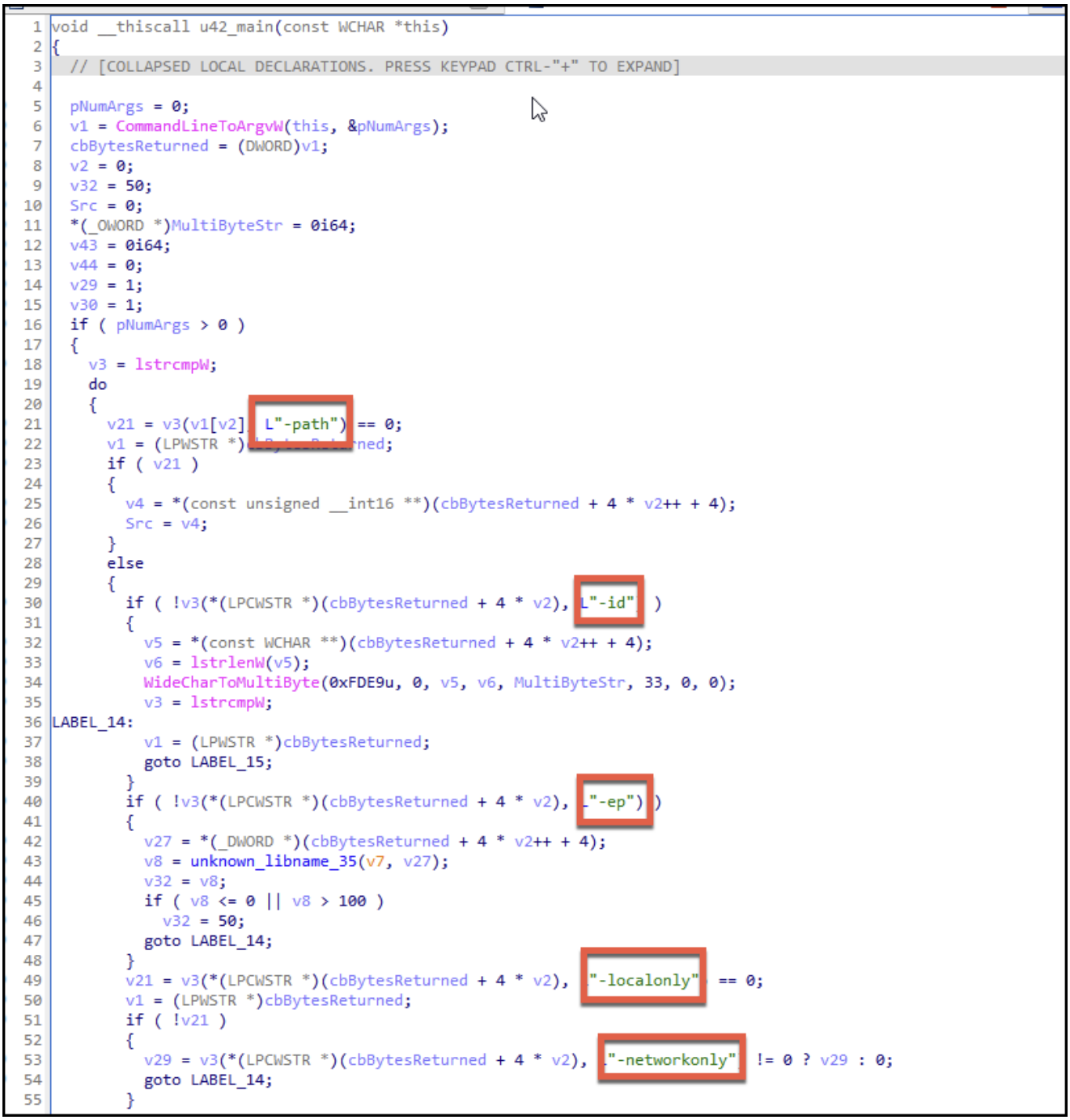
It is important to note that, while many ransomware families employ various forms of anti-analysis, as of late April, Royal ransomware does not employ anti-analysis tricks or string encryption.

There are five possible arguments for the Windows variant of Royal ransomware:

Argument	Purpose
-path	Path to be used for targeting encryption

-id	32-character ID for running sample
-ep	Encryption percentage - indicates the percentage of each file to be encrypted
-localonly	Encrypt only the local system
-networkonly	Encrypt file shares connected to system

In Figure 5 below, the decompiled view contains the various command-line arguments to be evaluated at the start of the binary being executed.



```
void __thiscall u42_main(const WCHAR *this)
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  pNumArgs = 0;
  v1 = CommandLineToArgvW(this, &pNumArgs);
  cbBytesReturned = (DWORD)v1;
  v2 = 0;
  v32 = 50;
  Src = 0;
  *(_OWORD *)MultiByteStr = 0i64;
  v43 = 0i64;
  v44 = 0;
  v29 = 1;
  v30 = 1;
  if ( pNumArgs > 0 )
  {
    v3 = lstrcmpW;
    do
    {
      v21 = v3(v1[v2], L"-path") == 0;
      v1 = (LPWSTR *)[illegible]rned;
      if ( v21 )
      {
        v4 = *(const unsigned __int16 **)(cbBytesReturned + 4 * v2++ + 4);
        Src = v4;
      }
      else
      {
        if ( !v3(*(LPCWSTR *)(cbBytesReturned + 4 * v2), L"-id") )
        {
          v5 = *(const WCHAR **)(cbBytesReturned + 4 * v2++ + 4);
          v6 = lstrlenW(v5);
          WideCharToMultiByte(0xFDE9u, 0, v5, v6, MultiByteStr, 33, 0, 0);
          v3 = lstrcmpW;
LABEL_14:
          v1 = (LPWSTR *)cbBytesReturned;
          goto LABEL_15;
        }
        if ( !v3(*(LPCWSTR *)(cbBytesReturned + 4 * v2), L"-ep") )
        {
          v27 = *(_DWORD *)(cbBytesReturned + 4 * v2++ + 4);
          v8 = unknown_libname_35(v7, v27);
          v32 = v8;
          if ( v8 <= 0 || v8 > 100 )
            v32 = 50;
          goto LABEL_14;
        }
        v21 = v3(*(LPCWSTR *)(cbBytesReturned + 4 * v2), L"-localonly") == 0;
        v1 = (LPWSTR *)cbBytesReturned;
        if ( !v21 )
        {
          v29 = v3(*(LPCWSTR *)(cbBytesReturned + 4 * v2), L"-networkonly") != 0 ? v29 : 0;
          goto LABEL_14;
        }
```

Figure 5. Command-line arguments Royal accepts as inputs.

Upon evaluating the command-line arguments provided, the ransomware then will create a cmd.exe process with the parameter to execute vssadmin delete shadows /all /quiet. The command is part of the standard ransomware playbook for impacting restoration services.

For the encryption process, Royal ransomware has a hard-coded RSA public key within the binary and uses AES for encryption. The AES encryption is set up using a 32-byte key and a 16-byte initialization vector (IV). The encrypted files are encrypted with the extension .royal_w.

During file enumeration and encryption, the sample avoids files with the following extensions and filenames (also shown in Figure 6):

Extensions:

- .exe
- .dll
- .bat
- .lnk
- .royal_w
- .royal_u

Files and folders:

- README.TXT
- Windows
- Royal
- Recycle.bin
- Google
- Perflogs
- Mozilla
- Tor browser
- Boot
- $Windows.~ws
- $Windows.~bt
- Windows.old

During the encryption process, if a file is encountered which is actively being used by the computer system, the ransomware can use the RestartManager API functionality to close a file. As shown in Figure 6, the strings related to skipped extensions and folder paths are shown in the .rdata section of the binary.

```
.rdata:0060A820 ; const wchar_t aExe
.rdata:0060A820 aExe:                                   ; DATA XREF: u42_wrapper_check_exclusions+D↑o
.rdata:0060A820                                         ; convert_to_stat_mode(int,wchar_t const * const)+4F↑o
.rdata:0060A820                 text "UTF-16LE", '.exe',0
.rdata:0060A82A                 align 4
.rdata:0060A82C aDll:                                   ; DATA XREF: u42_wrapper_check_exclusions:loc_44C480↑o
.rdata:0060A82C                 text "UTF-16LE", '.dll',0
.rdata:0060A836                 align 4
.rdata:0060A838 ; const wchar_t aBat
.rdata:0060A838 aBat:                                   ; DATA XREF: u42_wrapper_check_exclusions:loc_44C4FD↑o
.rdata:0060A838                                         ; convert_to_stat_mode(int,wchar_t const * const)+71↑o
.rdata:0060A838                 text "UTF-16LE", '.bat',0
.rdata:0060A842                 align 4
.rdata:0060A844 aLnk:                                   ; DATA XREF: u42_wrapper_check_exclusions:loc_44C57A↑o
.rdata:0060A844                 text "UTF-16LE", '.lnk',0
.rdata:0060A84E                 align 10h
.rdata:0060A850 aRoyalW:                                ; DATA XREF: u42_wrapper_check_exclusions:loc_44C5F7↑o
.rdata:0060A850                                         ; u42_wrapper_file_encryption+123↑o
.rdata:0060A850                 text "UTF-16LE", '.royal_w',0
.rdata:0060A862                 align 4
.rdata:0060A864 aRoyalU:                                ; DATA XREF: u42_wrapper_check_exclusions:loc_44C674↑o
.rdata:0060A864                 text "UTF-16LE", '.royal_u',0
.rdata:0060A876                 align 4
.rdata:0060A878 aReadmeTxt_0:                           ; DATA XREF: u42_wrapper_check_exclusions:loc_44C6F1↑o
.rdata:0060A878                 text "UTF-16LE", 'README.TXT',0
.rdata:0060A88E                 align 10h
.rdata:0060A890 aWindows:                               ; DATA XREF: u42_wrapper_check_exclusions:loc_44C76E↑o
.rdata:0060A890                 text "UTF-16LE", 'windows',0
.rdata:0060A8A0 aRoyal:                                 ; DATA XREF: u42_wrapper_check_exclusions:loc_44C7F4↑o
.rdata:0060A8A0                 text "UTF-16LE", 'royal',0
.rdata:0060A8AC aRecycleBin:                            ; DATA XREF: u42_wrapper_check_exclusions:loc_44C871↑o
.rdata:0060A8AC                 text "UTF-16LE", '$recycle.bin',0
.rdata:0060A8C6                 align 4
.rdata:0060A8C8 aGoogle:                                ; DATA XREF: u42_wrapper_check_exclusions:loc_44C8EE↑o
.rdata:0060A8C8                 text "UTF-16LE", 'google',0
.rdata:0060A8D6                 align 4
.rdata:0060A8D8 aPerflogs:                              ; DATA XREF: u42_wrapper_check_exclusions:loc_44C96B↑o
.rdata:0060A8D8                 text "UTF-16LE", 'perflogs',0
.rdata:0060A8EA                 align 4
.rdata:0060A8EC aMozilla:                               ; DATA XREF: u42_wrapper_check_exclusions:loc_44C9E8↑o
.rdata:0060A8EC                 text "UTF-16LE", 'mozilla',0
.rdata:0060A8FC aTorBrowser:                            ; DATA XREF: u42_wrapper_check_exclusions:loc_44CA65↑o
.rdata:0060A8FC                 text "UTF-16LE", 'tor browser',0
.rdata:0060A914 aBoot:                                  ; DATA XREF: u42_wrapper_check_exclusions:loc_44CAE2↑o
.rdata:0060A914                 text "UTF-16LE", 'boot',0
.rdata:0060A91E                 align 10h
.rdata:0060A920 aWindowsWs:                             ; DATA XREF: u42_wrapper_check_exclusions:loc_44CB5F↑o
.rdata:0060A920                 text "UTF-16LE", '$windows.~ws',0
.rdata:0060A93A                 align 4
.rdata:0060A93C aWindowsBt:                             ; DATA XREF: u42_wrapper_check_exclusions+7C3↑o
.rdata:0060A93C                 text "UTF-16LE", '$windows.~bt',0
.rdata:0060A956                 align 4
.rdata:0060A958 aWindowsOld:                            ; DATA XREF: u42_wrapper_check_exclusions+7E3↑o
.rdata:0060A958                 text "UTF-16LE", 'windows.old',0
```

Figure 6. Extensions and folders excluded.

The ransom note dropped as a README.txt is shown in Figure 7 below.

README.TXT

```
Hello!

    If you are reading this, it means that your system were hit by Royal ransomware.
    Please contact us via :
    http://royal2xthig3ou5hd7zsliqagy6yygk2cdelaxtni2fyad6dpmpxedid.onion/11111111111111111111111111111111

In the meantime, let us explain this case.It may seem complicated, but it is not!
Most likely what happened was that you decided to save some money on your security infrastructure.
Alas, as a result your critical data was not only encrypted but also copied from your systems on a secure server.
From there it can be published online.Then anyone on the internet from darknet criminals, ACLU journalists, Chinese government(different names for the same thing),
and even your employees will be able to see your internal documentation: personal data, HR reviews, internal lawsuitsand complains, financial reports, accounting, intellectual property, and more!

    Fortunately we got you covered!

Royal offers you a unique deal.For a modest royalty(got it; got it ? ) for our pentesting services we will not only provide you with an amazing risk mitigation service,
covering you from reputational, legal, financial, regulatory, and insurance risks, but will also provide you with a security review for your systems.
To put it simply, your files will be decrypted, your data restoredand kept confidential, and your systems will remain secure.

    Try Royal today and enter the new era of data security!
    We are looking to hearing from you soon!
```

Figure 7. Royal ransomware ransom note.

Royal also contains functionality to encrypt network shares connected to the victim machine. The decompiled view of the code path responsible for enumerating shares is shown in Figure 8 below.

```
_DWORD *__thiscall u42_wrapper_enumshare(_DWORD *this)
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  result = this;
  v2 = 512;
  v8 = this;
  v3 = this + 9;
  v9 = 512;
  v10 = this + 9;
  do
  {
    if ( !v3[1] )
    {
      memset(szAddressString, 0, sizeof(szAddressString));
      *(_DWORD *)&saAddress.sa_data[2] = *v3;
      *(_DWORD *)&saAddress.sa_family = 2;
      dwAddressStringLength = 44;
      WSAAddressToStringW(&saAddress, 0x10u, 0, szAddressString, &dwAddressStringLength);
      do
      {
        entriesread = 0;
        totalentries = 0;
        resume_handle = 0;
        bufptr = 0;
        result = (_DWORD *)NetShareEnum(
                             szAddressString,
                             1u,
                             &bufptr,
                             0xFFFFFFFF,
                             &entriesread,
                             &totalentries,
                             &resume_handle);
        v7 = result;
        if ( result && result != (_DWORD *)234 )
          break;
        v4 = (LPCWSTR *)bufptr;
        v5 = 1;
        if ( entriesread )
        {
          do
```

```
        {
          if ( lstrcmpiW(L"ADMIN$", *v4) )
          {
            if ( lstrcmpiW(L"IPC$", *v4) )
            {
              wsprintfW(Src, &off_60AA68, szAddressString, *v4);
              exclusion_check(v6, Src);
              sub_44B7B0(v6[0], (int)v6[1], (int)v6[2], (int)v6[3], (int)v6[4], (int)v6[5]);
            }
          }
          ++v5;
          v4 += 3;
        }
        while ( v5 <= entriesread );
        v4 = (LPCWSTR *)bufptr;
      }
      result = (_DWORD *)NetApiBufferFree(v4);
    }
    while ( v7 == (_DWORD *)234 );
    v3 = v10;
    v2 = v9;
  }
  v3 += 8;
  --v2;
  v10 = v3;
  v9 = v2;
 }
 while ( v2 );
 return result;
}
```

Figure 8. Enumerates network shares and excludes ADMIN and IPC shares.

For supporting cryptographic operations used in the ransomware, the code is statically compiled with OpenSSL. The cryptographic references can be seen in Figure 9 below and can be cross-referenced by examining the library on GitHub.

Address	Length	Type	String
.rdata:0059...	00000016	C	crypto\\rsa\\rsa_crpt.c
.rdata:0059...	00000015	C	crypto\\bio\\bio_lib.c
.rdata:0059...	00000017	C	crypto\\rand\\rand_lib.c
.rdata:0059...	00000015	C	crypto\\bio\\bss_mem.c
.rdata:0059...	00000013	C	crypto\\bn\\bn_lib.c
.rdata:0059...	00000013	C	crypto\\bn\\bn_ctx.c
.rdata:0059...	00000013	C	crypto\\bn\\bn_add.c
.rdata:0059...	00000013	C	crypto\\bn\\bn_gcd.c
.rdata:0059...	00000015	C	crypto\\bn\\bn_blind.c
.rdata:0059...	0000005D	C	C:\\Users\\User\\Desktop\\vcpkg\\buildtrees\\openssl\\x86-windows-static-rel\\crypto\\err\\err_local.h
.rdata:0059...	00000018	C	crypto\\err\\err_blocks.c
.rdata:0059...	00000015	C	crypto\\threads_win.c
.rdata:0059...	00000011	C	crypto\\ex_data.c
.rdata:0059...	00000016	C	crypto\\bio\\bio_sock.c
.rdata:0059...	00000011	C	crypto\\err\\err.c
.rdata:0059...	00000016	C	crypto\\bio\\bio_addr.c
.rdata:0059...	00000016	C	crypto\\bio\\bio_meth.c
.rdata:0059...	00000015	C	crypto\\stack\\stack.c
.rdata:0059...	0000000F	C	crypto\\o_str.c
.rdata:0059...	0000000E	C	crypto\\init.c
.rdata:0059...	00000011	C	crypto\\context.c
.rdata:0059...	00000010	C	crypto\\params.c
.rdata:0059...	00000016	C	crypto\\evp\\evp_rand.c
.rdata:0059...	00000018	C	crypto\\rand\\rand_pool.c
.rdata:0059...	00000014	C	crypto\\initthread.c
.rdata:0059...	00000017	C	crypto\\conf\\conf_lib.c
.rdata:0059...	00000017	C	crypto\\conf\\conf_mod.c
.rdata:0059...	00000018	C	crypto\\engine\\tb_rand.c
.rdata:0059...	00000019	C	crypto\\engine\\eng_init.c
.rdata:0059...	00000017	C	crypto\\buffer\\buffer.c
.rdata:0059...	00000016	C	crypto\\bio\\bss_file.c
.rdata:0059...	00000016	C	crypto\\evp\\p_legacy.c
.rdata:0059...	00000013	C	crypto\\evp\\p_lib.c
.rdata:0059...	00000014	C	crypto\\ec\\ec_asn1.c
.rdata:0059...	00000013	C	crypto\\ec\\ec_key.c
.rdata:0059...	00000015	C	crypto\\rsa\\rsa_lib.c
.rdata:0059...	00000014	C	crypto\\dh\\dh_asn1.c
.rdata:0059...	00000015	C	crypto\\dsa\\dsa_lib.c
.rdata:0059...	00000017	C	crypto\\x509\\x_pubkey.c
.rdata:0059...	00000014	C	crypto\\x509\\x_req.c
.rdata:0059...	00000014	C	crypto\\x509\\x_crl.c
.rdata:0059...	00000015	C	crypto\\pem\\pem_lib.c
.rdata:0059...	00000015	C	crypto\\pem\\pem_oth.c
.rdata:0059...	00000016	C	crypto\\pem\\pem_pkey.c
.rdata:0059...	00000024	C	crypto\\encode_decode\\encoder_meth.c
.rdata:0059...	00000023	C	crypto\\encode_decode\\encoder_lib.c
.rdata:0059...	00000024	C	crypto\\encode_decode\\encoder_pkey.c
.rdata:0059...	00000011	C	crypto\\mem_sec.c
.rdata:0059...	00000014	C	crypto\\bn\\bn_mont.c
.rdata:0059...	00000017	C	crypto\\bio\\bio_print.c
.rdata:0059...	00000013	C	crypto\\bn\\bn_div.c
.rdata:0059...	00000013	C	crypto\\bn\\bn_mod.c
.rdata:0059...	00000015	C	crypto\\bn\\bn_shift.c
.rdata:0059...	00000014	C	crypto\\bn\\bn_rand.c
.rdata:0059...	00000013	C	crypto\\bn\\bn_exp.c
.rdata:0059...	00000017	C	crypto\\bio\\bio_sock2.c
.rdata:0059...	00000015	C	crypto\\hash\\hash.c
.rdata:0059...	00000018	C	crypto\\engine\\eng_fat.c
.rdata:0059...	00000018	C	crypto\\engine\\eng_dyn.c
.rdata:0059...	00000018	C	crypto\\engine\\eng_lib.c
.rdata:005A...	00000019	C	crypto\\objects\\obj_dat.c

Figure 9. OpenSSL library statically compiled with ransomware binary.

Linux Variant

During development of this post, a Linux variant of Royal ransomware was identified by @BushidoToken on Feb. 1, 2023. This is the first known version not targeting Windows systems. However, considering many ransomware families have an ESXi/Linux focused

variant, this isn't unusual. It only makes sense that this group would expand their arsenal to impact other environments.

There are minimal differences between the Linux and Windows variants in terms of encryption. AES-256 is used for symmetric encryption, while RSA-4096 is used for asymmetric encryption within the sample.

Additionally, there is a lack of obfuscation within the Linux sample. All strings are stored as plaintext, including the RSA public key and the ransom note.

There are five possible arguments for the Linux variant of Royal ransomware:

Argument	Purpose
-id	32-character ID for running sample
-ep	Encryption percentage – indicates the percentage of each file that will be encrypted
-stopv	Indicates to the sample whether to stop VM-linked processes or not
-fork	Forks the current process for encryption
-logs	Informs the sample to log information to a file

During file enumeration and encryption, the sample avoids files with the following extensions and filenames:

Extensions:

- .v00
- .b00
- .sf
- .royal_u
- .royal_w
- .royal_log_
- .readme

The variant is also compiled with the OpenSSL library, resulting in a large number of unreferenced crypto-linked strings.

```
C       Montgomery Multiplication for x86_64, CRYPTOGAMS by <appro@openssl.org>
C       Montgomery Multiplication with scatter/gather for x86_64, CRYPTOGAMS by <appro@openssl.org>
C       GHASH for x86_64, CRYPTOGAMS by <appro@openssl.org>
C       Poly1305 for x86_64, CRYPTOGAMS by <appro@openssl.org>
C       rc4(8x,int)
C       rc4(8x,char)
C       rc4(16x,int)
C       RC4 for x86_64, CRYPTOGAMS by <appro@openssl.org>
C       Keccak-1600 absorb and squeeze for x86_64, CRYPTOGAMS by <appro@openssl.org>
C       SHA1 multi-block transform for x86_64, CRYPTOGAMS by <appro@openssl.org>
C       SHA1 block transform for x86_64, CRYPTOGAMS by <appro@openssl.org>
C       SHA256 multi-block transform for x86_64, CRYPTOGAMS by <appro@openssl.org>
C       SHA256 block transform for x86_64, CRYPTOGAMS by <appro@openssl.org>
C       AESNI-CBC+SHA1 stitch for x86_64, CRYPTOGAMS by <appro@openssl.org>
C       AESNI-CBC+SHA256 stitch for x86_64, CRYPTOGAMS by <appro@openssl.org>
C       AES for Intel AES-NI, CRYPTOGAMS by <appro@openssl.org>
C       Vector Permutation AES for x86_64/SSSE3, Mike Hamburg (Stanford University)
C       Camellia for x86_64 by <appro@openssl.org>
C       ChaCha20 for x86_64, CRYPTOGAMS by <appro@openssl.org>
C       X25519 primitives for x86_64, CRYPTOGAMS by <appro@openssl.org>
C       SHA512 block transform for x86_64, CRYPTOGAMS by <appro@openssl.org>
```

Figure 10. OpenSSL strings seen within Linux ransomware binary.

Conclusion

Royal ransomware has been more active this year, using a wide variety of tools and more aggressively targeting critical infrastructure organizations. Organizations should implement security best practices and be wary of the ongoing threat of ransomware. This is true not only for Royal ransomware but for other opportunistic criminal groups as well.

The Unit 42 team recommends that defenders have advanced logging capabilities deployed and configured properly. This includes tools such as Sysmon, Windows command-line logging and PowerShell logging.

Ideally, you should be forwarding these logs to a security information and event management tool (SIEM) to create queries and detection opportunities. Keep computer systems patched and up to date wherever possible to reduce the attack surface related to exploitation techniques.

Deploy an XDR/EDR solution to perform in-memory inspection and detect process injection techniques. Perform threat hunting looking for signs of unusual behavior related to security product defense evasion, service accounts for lateral movement and domain administrator-related user behavior.

Protections and Mitigations

Palo Alto Networks customers receive protections from the threats discussed above through the following products.

If you think you may have been compromised or have an urgent matter, get in touch with the Unit 42 Incident Response team or call:

- North America Toll-Free: 866.486.4842 (866.4.UNIT42)
- EMEA: +31.20.299.3130
- APAC: +65.6983.8730
- Japan: +81.50.1790.0200

Palo Alto Networks has shared these findings, including file samples and indicators of compromise, with our fellow Cyber Threat Alliance (CTA) members. CTA members use this intelligence to rapidly deploy protections to their customers and to systematically disrupt malicious cyber actors. Learn more about the Cyber Threat Alliance.

Indicators of Compromise

Hashes

595c869f8ec7eaf71fef44bad331d81bb934c886cdff99e1f013eec7acdaf8c9	Royal Windows Variant
b57e5f0c857e807a03770feb4d3aa254d2c4c8c8d9e08687796be30e2093286c	Royal Linux Variant
b64acb7dcc968b9a3a4909e3fddc2e116408c50079bba7678e85fee82995b0f4	Royal Linux Variant
b64acb7dcc968b9a3a4909e3fddc2e116408c50079bba7678e85fee82995b0f4	Royal Linux Variant
12a6d61b309171b41347d6795002247c8e2137522a756d35bb8ece5a82fc3774	Royal Linux Variant

Infrastructure

royal2xthig3ou5hd7zsliqagy6yygk2cdelaxtni2fyad6dpmpxedid[.]onion

Additional Resources

Get updates from Palo Alto Networks!

Sign up to receive the latest news, cyber threat intelligence and research from us

By submitting this form, you agree to our Terms of Use and acknowledge our Privacy Statement.