Conti Team One Splinter Group Resurfaces as Royal Ransomware with Callback Phishing Attacks

trendmicro.com/en_us/research/22/l/conti-team-one-splinter-group-resurfaces-as-royal-ransomware-wit.html

December 21, 2022

Ransomware

From September to December, we detected multiple attacks from the Royal ransomware group. In this blog entry, we discuss findings from our investigation of this ransomware and the tools that Royal ransomware actors used to carry out their attacks.

By: Ivan Nicole Chavez, Byron Gelera, Monte de Jesus, Don Ovid Ladores, Khristian Joseph Morales December 21, 2022 Read time: (words)

Royal ransomware may have been first observed by researchers around September 2022, but it has seasoned cybercriminals behind it: The threat actors running this ransomware — who used to be a part of Conti Team One, according to a mind map shared by Vitali Kremez — initially dubbed it Zeon ransomware, until they rebranded it to Royal ransomware. From September to December this year, we have detected multiple attacks from Royal ransomware, with the US and Brazil being the most targeted countries (Figure 1). This blog entry discusses in depth the findings from our investigation of samples of this new piece of ransomware, as well as the tools that Royal ransomware actors used to carry out their attacks.

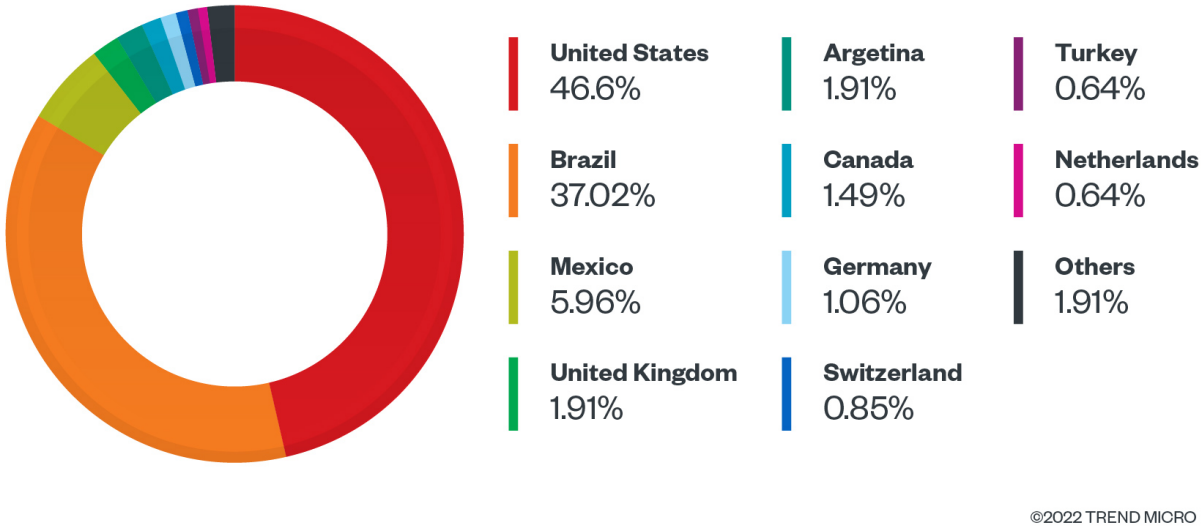

Figure 1. Percentage of Royal ransomware attacks by country

Infection Routine

External reports mention that the Royal ransomware group uses callback phishing as a means of delivering their ransomware to victims (Figure 2). These phishing attacks contain a number that leads to a service hired by the threat actors. When contacted, they will use social engineering tactics to lure victims into installing remote access software.

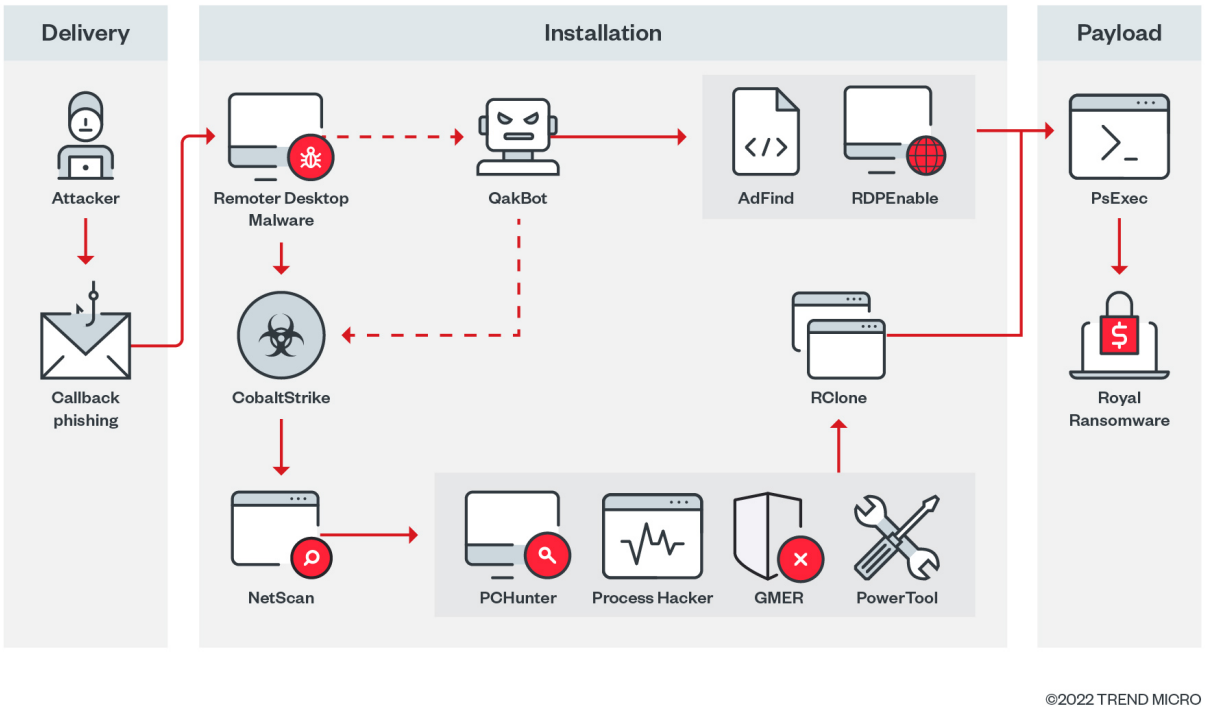

Figure 2. Royal ransomware's attack flow

Installation

Our investigation found that the ransomware actors used a compiled remote desktop malware, which was used to drop the tools they needed to infiltrate the victim's system: they used QakBot and Cobalt Strike for lateral movement, while NetScan was used to look for any remote systems connected to the network. Once they infiltrated the system, the ransomware actors used tools such as PCHunter, PowerTool, GMER, and Process Hacker to disable any security-related services running in the system. They then exfiltrate the victim's data via the RClone tool. We also observed an instance in which they used AdFind to look for active directories, then executed RDPEnable on the infected machine.

Payload

Once everything has been set up, the ransomware actors used PsEXEC to execute the malware. The PsEXEC commands contain the ID of the victim, along with any argument that the actors applied to the ransomware. There were also instances of the malware actors using PsEXEC to enable the remote desktop protocol (RDP) of a target system before executing the ransomware.

Analysis

In part of our analysis, we used a ransomware sample with the detection name *Ransom.Win64.YORAL.SMYXCJCT*. As shown in Table 1, Figure 3, and Figure 4, Royal ransomware requires an argument of "*-id {32-byte characters}*" to execute on a victim's machine. It also accepts "*-path*" to specify a target file for encryption and "*-ep {value}*" to calculate the partial file encryption of large files.

In some earlier samples of the ransomware, the binary wouldn't parse all the arguments due to a bug in the code. For example, "*-path*" won't be processed if provided after the "*-id*" argument; if provided before, there will be no "*-id*" argument, so it will not proceed.

Argument	Description
-path {target path}	If provided, will only encrypt the contents of the target path
-id {32-byte characters}	Will be used as the victim's ID, which will be appended on the TOR link found in the dropped ransom note. The process exists if not provided or if provided characters is not 32 bytes long
-ep	This argument is for the full or partial encryption of file routine

Table 1. Arguments accepted by the Royal ransomware binary

```
pNumArgs = 0;
v4 = GetCommandLineW();
v5 = CommandLineToArgvW(v4, &pNumArgs);
v6 = 50;
v7 = 0i64;
v8 = 0;
v21 = 0;
*MultiByteStr = 0i64;
for ( i = 0i64; v8 < pNumArgs; ++v5 )
{
  if ( lstrcmpW(*v5, L"-path") )
  {
    if ( lstrcmpW(*v5, L"-id") )
    {
      if ( !lstrcmpW(*v5, L"-ep") )
      {
        v11 = v5[1];
        ++v5;
        ++v8;
        v6 = unknown_libname_21(v11);
        if ( (v6 - 1) > 0x63 )
          v6 = 50;
      }
    }
    else
```

Figure 3. Arguments accepted by the ransomware binary

```
if ( lstrlenA(MultiByteStr) != 32 )
  ExitProcess(0);
```

Figure 4. Checking if length of provided “-id” is 32 bytes

It enumerates files and directories for encryption using FindFirstFileW, FindNextFileW, and FindClose APIs (Figure 5).

```
sub_14007CB80(&lpFileName, (a1 + 744), L"\\*");
v20 = &lpFileName;
if ( v27 >= 8 )
  v20 = lpFileName.m128i_i64[0];
v21 = FindFirstFileW(v20, (a1 + 56));
*(a1 + 6) = v21;
if ( v21 != -1i64 )
  traverse_14007BA80(a1);
if ( v27 >= 8 )
{
```

Figure 5. File enumeration

The ransomware looks for available network shares for network encryption by listing accessible local IPs, then uses NetShareEnum and attempts to connect on ADMIN$ and IPC$ shares (Figure 6).

```
WSAAddressToStringW(&saAddress, 0x10u, 0i64, szAddressString, &dwAddre
do
{
  entriesread = 0;
  totalentries = 0;
  resume_handle = 0;
  bufptr = 0i64;
  v4 = NetShareEnum(szAddressString, 1u, &bufptr, 0xFFFFFFFF, &entries
  v5 = v4;
  if ( v4 && v4 != 234 )
    break;
  v6 = bufptr;
  v7 = 1;
  if ( entriesread )
  {
    do
    {
      if ( lstrcmpiW(L"ADMIN$", *v6) && lstrcmpiW(L"IPC$", *v6) )
      {
        wsprintfW(&v24, L"\\\\%s\\%s", szAddressString, *v6);
        v11.m128i_i64[0] = 0i64;
        v9 = -1i64;
        v12 = 0i64;
```

Figure 6. Looking for accessible local IPs then trying to connect to ADMIN$ and IPC$

It checks for the number of processors in the infected system and uses it as a base for the concurrent running threads for file encryption, as shown in Figure 7. By doing so, Royal ransomware significantly increases the speed of its file encryption process.

```
GetNativeSystemInfo(&SystemInfo);
result = 2 * SystemInfo.dwNumberOfProcessors;
*(lpParameter + 530) = a2;
*(lpParameter + 524) = result;
v5 = 0;
if ( result )
{
  v6 = lpParameter + 48;
  do
  {
    result = CreateThread(0i64, 0i64, sub_14007F870, lpParameter, 0, 0i64);
    *v6 = result;
    ++v5;
    ++v6;
  }
  while ( v5 < *(lpParameter + 524) );
}
```

Figure 7. Checking the number of processors

Royal ransomware inhibits system recovery by deleting shadow copies (Figure 8) through the following command:

C:\\Windows\\System32\\vssadmin.exe delete shadows /all /quiet

```
wsprintfW(CommandLine, L" delete shadows /all /quiet");
StartupInfo.cb = 104;
HIDWORD(StartupInfo.hStdError) = 0;
*&ProcessInformation.dwProcessId = 0i64;
*(&StartupInfo.cb + 1) = 0i64;
*(&StartupInfo.lpDesktop + 4) = 0i64;
*&StartupInfo.dwY = 0i64;
*&StartupInfo.dwYCountChars = 0i64;
*(&StartupInfo.cbReserved2 + 1) = 0i64;
*(&StartupInfo.hStdInput + 4) = 0i64;
*&ProcessInformation.hProcess = 0i64;
if ( CreateProcessW(
       L"C:\\Windows\\System32\\vssadmin.exe",
       CommandLine,
       0i64,
       0i64,
       0,
       0,
       0i64,
       0i64,
       &StartupInfo,
       &ProcessInformation) )
{
  WaitForSingleObject(ProcessInformation.hProcess, 0x2710u);
```

Figure 8. Using vssadmin.exe to delete shadow copies

The ransomware encrypts files using OpenSSL's Advanced Encryption Standard (AES). It will encrypt the AES key and IV with RSA encryption using the embedded RSA public key (Figure 9). The RSA-encrypted AES key and IV will be appended on each encrypted file (Figure 10).

```
sub_140081240(
  v3,
  "-----BEGIN RSA PUBLIC KEY-----\n"
  "[redacted]\n"
  "[redacted]\n"
  "[redacted]\n"
  "[redacted]\n"
  "[redacted]\n"
  "[redacted]\n"
  "[redacted]\n"
  "[redacted]\n"
  "[redacted]\n"
  "[redacted]\n"
  "-----END RSA PUBLIC KEY-----\n"
  "\r\n",
  v4);
```

Figure 9. An RSA public key

```
generate_random_1400819B0(aes_key, 32);
generate_random_1400819B0(&aes_iv, 16);
v40[0] = aes_key[0];
v40[1] = aes_key[1];
v40[2] = aes_iv;
(RSAEncrypt_14007FE30)(48i64, v40, v40, a2, 4);
```

Figure 10. Generation of AES Key and IV

The malicious actors behind Royal ransomware use a form of intermittent encryption tactic to speed their encryption process: the ransomware first checks if the file size is divisible by 16, which is a requirement for AES (Figure 11). If not, it rounds up the total size until it is divisible by 16. For example, if the size is 18, it will append zero bytes to the file until it has a size of 32, which is now divisible by 16. Aside from appending the needed zero bytes, it also appends an extra 0x210 Zero bytes as a placeholder for the appended RSA encrypted key.

```
v8 = 0i64;
if ( a3.QuadPart % 16 > 0 )
  v8 = 16i64;
v31.QuadPart = v8 + 16 * (a3.QuadPart / 16);
v9 = v8 + 16 * (a3.QuadPart / 16 + 33) - a3.QuadPart;
v10 = (char *)operator new(v9);
if ( SetFilePointerEx(hObject, a3, 0i64, 0) )
{
  v11 = 0;
  v12 = 0;
  while ( 1 )
  {
    NumberOfBytesWritten = 0;
    if ( !WriteFile(hObject, &v10[v11], v9 - v12, &NumberOfBytesWritten, 0i64) || !NumberOfBytesWritten )
      break;
    v12 += NumberOfBytesWritten;
    v11 = v12;
    if ( (_DWORD)v9 == v12 )
    {
      SetEndOfFile(hObject);
      j_j_free(v10);
```

Figure 11. Royal ransomware checking if file size is divisible by 16

For a file size that has been rounded-up, Royal ransomware will check if the size is less than or equal to 5,245,000 bytes or if the value is set to 100 (0x64), as shown in Figure 12. If the file size is within these limits, it will encrypt the entire file. For files greater than 5,245,000 bytes, file encryption will take place per certain calculated blocks: for example, it will encrypt first N bytes, then skip the next N bytes, then encrypt the next N bytes, and so on.

```
if ( fSize.QuadPart <= 5245000 || X == 0x64 )
{
  v34 = 1;
  encrypt_block = fSize;
  X = 0x64i64;
}
else
{
  v34 = 10;
  v13 = fSize.LowPart / 100.0;
  encrypt_block.QuadPart = (X / 10.0 * v13) & 0xFFFFFFF0;
  liDistanceToMove.QuadPart = ((100.0 - X) / 10.0 * v13) & 0xFFFFFFF0;
}
```

Figure 12. Encryption process and calculation

Its calculation of N bytes is as follows:

> X / 10* (Original file size) & 0xFFFFFFF0

- where X is the value set before encryption
- X is either 0x32 (50) or 0x64 (100)
- This value will also be used as indicator if full encryption or partial encryption will be performed on the file

For example, with a file with a file size equal to 5,245,000:

> N = 50/10 * (5245000 / 100) & 0xFFFFFFF0 = 0x40060 (262240)

If the calculated N is greater than 1,024,000, it will simply encrypt per 1,024,000 block instead (Figure 13).

```
if ( v14.QuadPart - v15 > 1024000 )
  v17 = 1024000;
```

Figure 13. Condition if N is greater than 1,024,000

The encrypted file's structure would then be as follows (Table 2):

Description	Size
Encrypted File Contents	Rounded-up file size divisible by 16
RSA Encrypted Key	0x200 bytes
Size of encrypted file / offset address of RSA Encrypted Key	8 bytes
X value, 0x64 or provided value (usually 0x32), indicator if full or partial encryption	8 bytes

Table 2. An encrypted file's structure

The ransomware then renames the encrypted files by appending them with the ".royal" extension, as demonstrated in Figures 14 and 15.

```
{
  sub_14007CB80(v16, v15, L".royal");
  sub_14007C970(v16);
  v12 = sub_14007C970(v15);
  MoveFileExW(v12, v13, 8u);
  unknown_libname_4(v16);
}
```

Figure 14. Royal ransomware appending ".royal" to encrypted files

Figure 15. Encrypted files appended with the ".royal" extension

For each directory it traverses, Royal ransomware drops a text file named “README.TXT” that contains the ransom note (Figure 16), as well as an advertisement for its “pentesting services” that the ransomware actors will allegedly provide once the ransom has been paid (Figure 17).

```
sub_14007C880(lpFileName, a2, L"\\README.TXT");
v3 = lpFileName;
if ( v11 >= 8 )
  v3 = lpFileName[0];
v4 = CreateFileW(v3, 0x40000000u, 0, 0i64, 2u, 0, 0i64);
if ( v4 == -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_140078860(
       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"
```

Figure 16. Creation of the “README.TXT” file

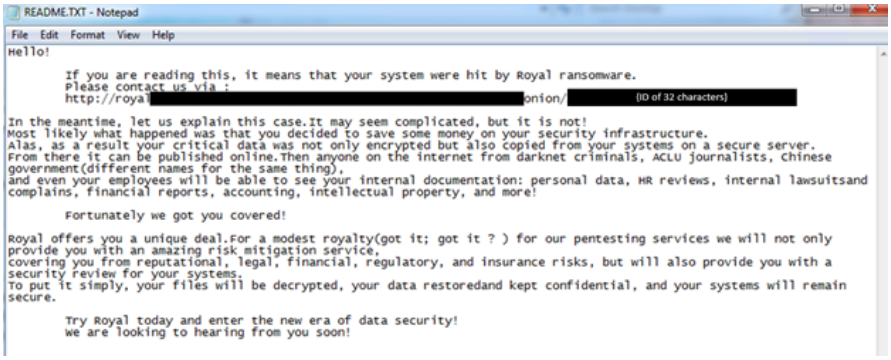

Figure 17. Contents of "README.TXT" with the sample ID we used appended on the TOR link.

Security Recommendations

Our investigation into Royal ransomware attacks shows how the group employs a mixture of both old and new techniques, which indicates that it is no newcomer to the ransomware scene. Their use of callback phishing to lure victims into installing remote desktop malware allows them to infiltrate the victim’s machine with relative ease. Their intermittent encryption tactics also hasten their encryption of a victim’s files, with the added benefit of evading detection measures that focus on looking for heavy file IO operations. Despite their “late” entry to the scene in September, the group already has ransomed multiple companies, and we expect them to be more active in the upcoming months. More details on Royal ransomware’s other capabilities can be found in Trend Micro’s Threat Encyclopedia.

We highly advise users and organizations to update their systems with the latest patches and apply multi-layered defense mechanisms. The emergence and success of the Royal ransomware gang underscore how ransomware actors are finding more innovative ways to repurposing existing tools and tactics as a means of augmenting their attacks. End users and enterprises alike can mitigate the risk of infection from new threats like Royal ransomware by following these security best practices:

- Enable multifactor authentication (MFA) to prevent attackers from performing lateral movement inside a network.
- Adhere to the 3-2-1 rule when backing up important files. This involves creating three backup copies on two different file formats, with one of the copies stored in a separate location.
- Patch and update systems regularly. It’s important to keep operating systems and applications up to date and maintain patch management protocols that can deter malicious actors from exploiting any software vulnerabilities.

Companies can also benefit from the use of multilayered detection and response solutions such as Trend Micro Vision One™, which provides powerful XDR capabilities that collect and automatically correlate data across multiple security layers — email, endpoints, servers, cloud workloads, and networks — to prevent attacks via automated protection, while also ensuring that no significant incidents go unnoticed. Trend Micro Apex One™ also provides next-level automated threat detection and response to protect endpoints against advanced issues, like human-operated ransomware.

Indicators of Compromise (IOCs)

SHA-256	Detection	Description

c0063d24f3de4e7b89abf9b690a3d264efc6ab7a626f73ad9f42d6bffe52bce7	Trojan.Win64.COBALT.BE	CobaltStrike
fef79160f0ce9aa9dec15c914f2c2b40b2ae1ec2b0e65e414545dbc994afd73d	Trojan.Win64.COBALT.BE	CobaltStrike
3434271f2038afaddad4caad8000e390b3573b2b53e02841653a4ee0dfd73674	Trojan.Win64.COBALT.BE	CobaltStrike
0ac0b3758359855e96367b6c83b0aabdc6cfb59b4caa1cec48632defd21cdf3c	Trojan.Win64.COBALT.BE	CobaltStrike
451cef0085dc5b474cc5c68af079d0367d7d2ec73ae2210788beb5297e1fbd6d	Trojan.Win64.COBALT.BE	CobaltStrike
e710e902507ad63e1d2ce1220212b1a751b70504259457234103bb22845a9424	Trojan.Win32.QAKBOT.DRSV	QakBot
2718dcbb503b6334078daf4af61e17a547fb80c9b811c26cfc9d32f5ce63a826	Trojan.Win32.QAKBOT.DRTE	QakBot
abf937fb2f162d1dbbe76c7386c9892db5191e17de586f0a5c49819cd68b5e0f	Trojan.Win32.DEYMA.AM	Compiled Remote Desktop Malware
bd2c2cf0631d881ed382817afcce2b093f4e412ffb170a719e2762f250abfea4	PUA.Win64.ProcHack.AC	Process Hacker
572d88c419c6ae75aeb784ceab327d040cb589903d6285bbffa77338111af14b	HackTool.Win32.NetScan.AG	NetScan
094d1476331d6f693f1d546b53f1c1a42863e6cde014e2ed655f3cbe63e5ecde	HackTool.Win32.ToolPow.SM	PowerTool
e8a3e804a96c716a3e9b69195db6ffb0d33e2433af871e4d4e1eab3097237173	PUA.Win32.GMER.YABBI	GMER
d1aa0ceb01cca76a88f9ee0c5817d24e7a15ad40768430373ae3009a619e2691	PUA.Win64.PCHunter.B	PCHunter
bb48f5c915ab7bbbbbf092a20169aaf3ced46b492ed69550854a55254ce10572	Backdoor.Win32.SWRORT.YXCJ5Z	Malware Component
e263b9d5467bf724000966da2acfe06520a464c566e4b3d9833213f850f3f1f2	HackTool.Win32.Adfind.THLOFBB	AdFind
ac49c114ef137cc198786ad8daefa9cfcc01f0c0a827b0e2b927a7edd0fca8b0	HackTool.BAT.RDPEnable.A	RDPEnable
2598e8adb87976abe48f0eba4bbb9a7cb69439e0c133b21aee3845dfccf3fb8f	Ransom.Win64.YORAL.SMYXCJCT	Royal Ransomware Binary
cdd7814074872fc35d18740cdd4e8a5fefcfd6b457fde2920383fd5b11903fc5	Ransom_Royal.R06CC0DK222	Royal Ransomware Binary
a61b71ee73ea8c0f332591e361adeda04705c65b5f4d549066677ec4e71212f7	Ransom.Win32.YORAL.YXCKB	Royal Ransomware Binary
56e8bd8b0c5bfb87956f7915bc47a9ecf5d338b804cee1dccacf53400d602be3	Ransom.Win32.YORAL.YECJYT	Royal Ransomware Binary