STREAMSCAN

ANALYSE

RANCONGICIEL SPACECOLON

JAN 2025 //

Introduction

Le présent rapport concerne l'analyse d'un binaire malicieux qui se trouve être un ransomware. Les détails sur le fichier sont :

- Nom : **auto.exe**
- Hash SHA 1 = 28d0d07c190fd9827560d3e59dd70a8a26392f82
- Hash MD5 : c6c413543a540df9e8bba36aae7518e1
- Taille: 6.37 MB (6674944 bytes)

L'objectif de l'analyse (rétro-ingénierie de code malicieux) est de mieux comprendre ce rançongiciel ainsi que son comportement. Nos analyses lient ce ransomware au groupe Space Colon. Il se peut aussi qu'il soit distribué sur d'autres noms.

Analyse statique du ransomware

Ce ransomware est une application GUI écrite avec Delphi. D'après nos recherches, l'échantillon le plus ancien de ce ransomware est apparu en mars 2023.

La manière la plus simple de traiter les fichiers binaires Delphi est d'utiliser Interactive Delphi Reconstructor (IDR), qui analyse avec précision les structures de données internes et les types de fichiers binaires. Il permet également d'inspecter l'interface graphique sans exécuter l'application.

Le point d'entrée de l'interface graphique serait l'événement Form1.OnCreate qui est commun aux applications basées sur l'interface graphique. Par défaut, la fonction S & E, qui signifie rechercher et crypter, procède automatiquement au cryptage.

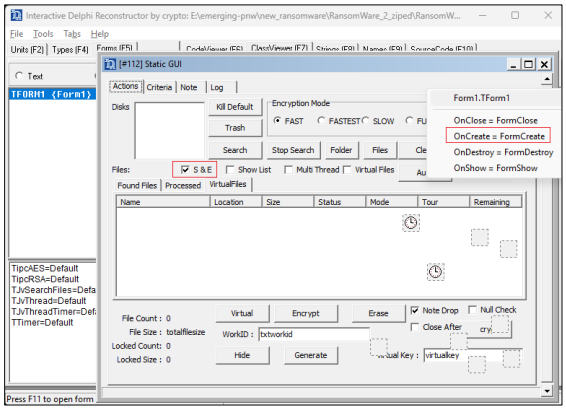


Figure 1: En utilisant l'inspection sans tête de l'IDR, nous pouvons visualiser l'interface graphique et les propriétés et antécédents de ses composants.

Initialisation :

Pendant l'initialisation du ransomware, la note de rançon est décodée en mémoire à l'aide du schéma de codage base64. Plus tard, un fichier nommé ".id_rsa" situé dans "C:\NUsers\NDesertSun" qui est

potentiellement la machine de l'acteur de la menace, cependant, aucune corrélation n'a été trouvée en rapport avec le nom d'utilisateur "DesertSun". Le nom de fichier ".id_rsa" est un nom par défaut utilisé par SSH lors de la génération d'une paire de clés. Comme il n'existe pas sur le système, il est simplement ignoré pendant l'exécution, mais pendant le débogage, il soulève une exception qui provoque le plantage du programme.

```
(*(*v3 + 28))(v3, &v23, *v3);                 // get encoded ransom note
r_decode_ransom_note(v23, &v24, v3, a2, a3);  // decoded ransom note
(*(*v3 + 48))(v3, v24);
v4 = sub_475380(v16);
ExceptionList = &savedregs;
v14[1] = &loc_6F7CDF;
v14[0] = NtCurrentTeb()->NtTib.ExceptionList;
__writefsdword(0, v14);
(*(v4->load_from_file + 27))(v4, L"c:\\Users\\DesertSun\\.id_rsa");// load_from_file
__writefsdword(0, v14[0]);
dword_7F97B0 = sub_6FE63C(v16, &byte_6EF438, -1LL);
LOBYTE(v5) = 1;
dword_7F97AC = sub_6FE008(-1LL, &byte_6EC0F4, v5, 10);// TThreadedQueue<System.String>
LOBYTE(v6) = 1;
dword_7F97B8 = sub_5472EC(VMT_546654_TCriticalSection, v6);
n_threads = sub_6FEFA0(v16);
sub_57E758(*(v26 + 1084), &v22);              // TForm1.SizeMinUnit:TComboBox
ExceptionList = v22;
TGroupBox_GetCaption(*(v26 + 1072), &v21);    // 100 - Atleast
v7 = r_str_to_int(v21);
v8 = r_size_to_byte_size(v7, ExceptionList);
r_set_min_size(*(*(v26 + 1300) + 784), SHIDWORD(v8), v9, SHIDWORD(v8), v8);
```

Figure 2: Form1.OnCreate - Décode la note et vérifie l'existence de "C:\NUsers\NDesertSun\N.id_rsa" et initialise les fils de discussion pour la création.

Outre l'initialisation, le ransomware a la capacité de charger un fichier personnalisé "note.txt" et des extensions à chiffrer nommées "ext.txt". Il tue également divers processus et services susceptibles de verrouiller les fichiers et d'empêcher le chiffrement des fichiers critiques. Enfin, les dossiers de la corbeille de tous les lecteurs sont effacés du lecteur "A" au lecteur "Z".

```
if ( r_is_file_exist(L"note.txt", 1) )
  (*(**(*(v26 + 1008) + 728) + 108))(*(*(v26 + 1008) + 728), L"note.txt");
r_is_file_exist(L"ext.txt", 1);
r_show_process_services_list(v26);
kernel32_Sleep_1(0x3E8u);
r_clear_recycle_bin(v3);
kernel32_Sleep_1(0x3E8u);
```

Figure 3: Autres routines trouvées dans la fonction d'initialisation

```
006FD2B6    E8 DDBED2FF     call auto.429198
006FD2BB    8B85 88FDFFFF   mov eax,dword ptr ss:[ebp-278]   [ebp-278]:L"x32dbg.exe"
006FD2C1    E8 2A52FFFF     call auto.6F24F0
006FD2C6    A1 44827100     mov eax,dword ptr ds:[718244]   eax:L"x32dbg.exe"
006FD2CB    8B00            mov eax,dword ptr ds:[eax]      eax:L"x32dbg.exe"
006FD2CD    E8 1A61F4FF     call auto.6433EC
006FD2D2    33C0            xor eax,eax                     eax:L"x32dbg.exe"
```

Figure 4: Empêche le débogage du processus car il met fin à tous les processus qui ne figurent pas dans la liste des processus autorisés.

```
if ( v1 )
{
  hProcess = kernel32_OpenProcess(1u, 0, v11[3]);
  if ( hProcess )
  {
    v4 = &savedregs;
    v3[1] = &loc_6F25F3;
    v3[0] = NtCurrentTeb()->NtTib.ExceptionList;
    __writefsdword(0, v3);
    kernel32_TerminateProcess(hProcess, 0);
    __writefsdword(0, v3[0]);
    v4 = &loc_6F25FA;
    kernel32_CloseHandle_0(hProcess);
  }
}
```

Figure 5: Le pseudocode montre que l'énumération et l'arrêt des processus

```
v9 = advapi32_OpenServiceW(v7, v8, v12);
if ( v9 )
{
  if ( advapi32_ControlService(v9, SERVICE_CONTROL_STOP, &ServiceStatus)
    && advapi32_QueryServiceStatus(v9, &ServiceStatus) )
  {
    do
    {
      if ( ServiceStatus.dwCurrentState == 1 )
        break;
      dwCheckPoint = ServiceStatus.dwCheckPoint;
      kernel32_Sleep_1(0x1388u);
      if ( !advapi32_QueryServiceStatus(v9, &ServiceStatus) )
        break;
    }
    while ( dwCheckPoint <= ServiceStatus.dwCheckPoint );
    v16 = ServiceStatus.dwCurrentState == 1;
  }
  advapi32_CloseServiceHandle(v9);
}
advapi32_CloseServiceHandle(v7);
```

Figure 6: Enumération et arrêt d'un service à l'aide de l'API ControlService avec le code de contrôle SERVICE_CONTROL_STOP

```
LOWORD(a1) = 'A';
do
{
  (unknown_libname_87)(&v5, a1);
  r_concatenate_string(&v6, v5, L":\\$Recycle.Bin\\");
  LOBYTE(v1) = 1;
  if ( r_is_dir_exist(v6, v1) )
    r_clear_recycle_bin(v6);
  ++a1;
}
while ( a1 != '[' ); // Checks until 'Z' before '['
```

Figure 7: Les dossiers de la corbeille sur tous les lecteurs sont énumérés de "A" à "Z" et l'effacent.

```
006F26B2    8945 EA          mov dword ptr ss:[ebp-16],eax     [ebp-16]:L"C:\\$Recycle.Bin\\"
006F26B5    33C0             xor eax,eax
006F26B7    8945 EE          mov dword ptr ss:[ebp-12],eax
006F26BA    66:C745 F2 1406  mov word ptr ss:[ebp-E],614
006F26C0    33C0             xor eax,eax
006F26C2    8945 FC          mov dword ptr ss:[ebp-4],eax
006F26C5    8D45 E2          lea eax,dword ptr ss:[ebp-1E]
006F26C8    50               push eax
006F26C9    E8 628BE5FF      call <JMP.&SHFileOperationW>
```

Figure 8: Utilisation de SHFileOperationW avec la commande de suppression de fichier

```
006F9F67    8D55 E0          lea edx,dword ptr ss:[ebp-20]     [ebp-20]:L"*.mdf;*.ldf;*.ndf;*.dbf;*.db;*.dbs;
006F9F6A    8B83 FC040000    mov eax,dword ptr ds:[ebx+4FC]
006F9F70    E8 37CE6FFF      call auto.55ABAC
006F9F75    8B55 E0          mov edx,dword ptr ss:[ebp-20]     [ebp-20]:L"*.mdf;*.ldf;*.ndf;*.dbf;*.db;*.dbs;
006F9F78    8B83 14050000    mov eax,dword ptr ds:[ebx+514]
006F9F7E    8B80 10030000    mov eax,dword ptr ds:[eax+310]
006F9F84    E8 AB40FEFF      call auto.6DE034
006F9F89    EB 16            jmp auto.6F9FA1
006F9F8B    8B83 14050000    mov eax,dword ptr ds:[ebx+514]
006F9F91    8B80 10030000    mov eax,dword ptr ds:[eax+310]
edx=03637E7C L"*.mdf;*.ldf;*.ndf;*.dbf;*.db;*.dbs;*.log1;*.dat;*.mdb;*.ora;*.fdb;*.wt;*.gdb;*.bak;*.backup;*.vib;*.v
auto.006FA08C
```

Figure 9: Analyse la liste intégrée des extensions de fichiers cibles

Chiffrement :

La ransomware utilise la bibliothèque IPWorks Encrypt, qui met en œuvre un cryptage puissant à l'aide des principales normes cryptographiques, intégrées dans sa ressource qui peut être attribuée à la Turquie selon certaines sources non confirmées.

```
module = r_load_module(result);
if ( module )
{
  IPWorksEncrypt_EvtStr = r_get_proc_address_(module, "IPWorksEncrypt_EvtStr");
  IPWorksEncrypt_Stream = r_get_proc_address_(module, "IPWorksEncrypt_Stream");
  IPWorksEncrypt_RSA_Create = r_get_proc_address_(module, "IPWorksEncrypt_RSA_Create");
  IPWorksEncrypt_RSA_Destroy = r_get_proc_address_(module, "IPWorksEncrypt_RSA_Destroy");
  IPWorksEncrypt_RSA_Set = r_get_proc_address_(module, "IPWorksEncrypt_RSA_Set");
  IPWorksEncrypt_RSA_Get = r_get_proc_address_(module, "IPWorksEncrypt_RSA_Get");
  IPWorksEncrypt_RSA_GetLastError = r_get_proc_address_(module, "IPWorksEncrypt_RSA_GetLastError");
  IPWorksEncrypt_RSA_GetLastErrorCode = r_get_proc_address_(module, "IPWorksEncrypt_RSA_GetLastErrorCode");
  IPWorksEncrypt_RSA_SetLastErrorAndCode = r_get_proc_address_(module, "IPWorksEncrypt_RSA_SetLastErrorAndCode");
  IPWorksEncrypt_RSA_GetEventError = r_get_proc_address_(module, "IPWorksEncrypt_RSA_GetEventError");
  IPWorksEncrypt_RSA_GetEventErrorCode = r_get_proc_address_(module, "IPWorksEncrypt_RSA_GetEventErrorCode");
  IPWorksEncrypt_RSA_SetEventErrorAndCode = r_get_proc_address_(
                                              module,
                                              "IPWorksEncrypt_RSA_SetEventErrorAndCode");
  IPWorksEncrypt_RSA_CheckIndex = r_get_proc_address_(module, "IPWorksEncrypt_RSA_CheckIndex");
  IPWorksEncrypt_RSA_Do = r_get_proc_address_(module, "IPWorksEncrypt_RSA_Do");
}
```

Figure 10: Les API d'IPWorks récoltées et utilisées pour le chiffrement

-----BEGIN RSA PUBLIC KEY-----
\r\nMIGJAoGBANaTwGg0JUiuyRgdxFnDq89RxgtkkxIF0cZ4JLqg
M7zPlY5Agh9r2rk\rnl8DR0aneOr+0WPBgmJiYN33fdCty1FMzan
x/WVfuENyamjZPyVqy4SS4KPLdwsZw\r\nSAkGNCieKSO4tEA
mIlFs4Gz+bG4FoSO1FU50BqRIK99ccSiro4MTAgMBAAE=\r\rn--
---END RSA PUBLIC KEY-----\r\n

Figure 11: Clé publique RSA

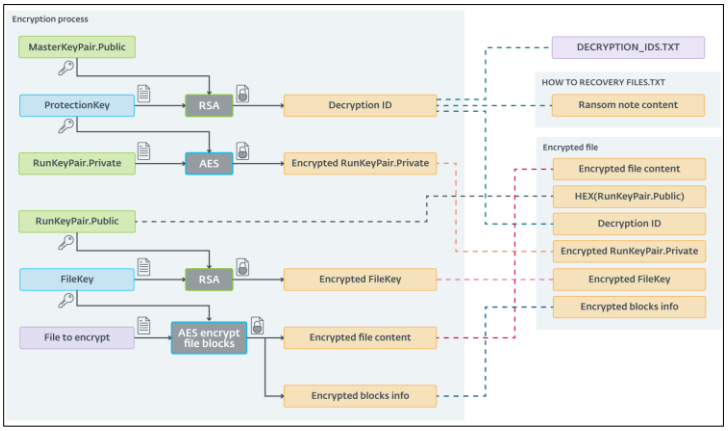


Figure 12: Schéma de cryptage complexe utilisé par Space Colon.

Figure 13: Modes de cryptage

Cinq modes de cryptage sont disponibles, et la plupart d'entre eux fonctionnent de la même manière, mais avec des types différents de cryptage des fichiers.

- Complet - cryptage complet
- Le plus rapide - crypte le premier 1% de la taille totale du fichier
- Cryptage rapide - intermittent
- Cryptage lent - intermittent

- Effacement - d'après l'échantillon binaire, il semble que cette fonction ne fonctionne pas comme prévu. Toutefois, si elle est utilisée, certaines parties du fichier sont remplacées par une constante qui le rend irrécupérable.

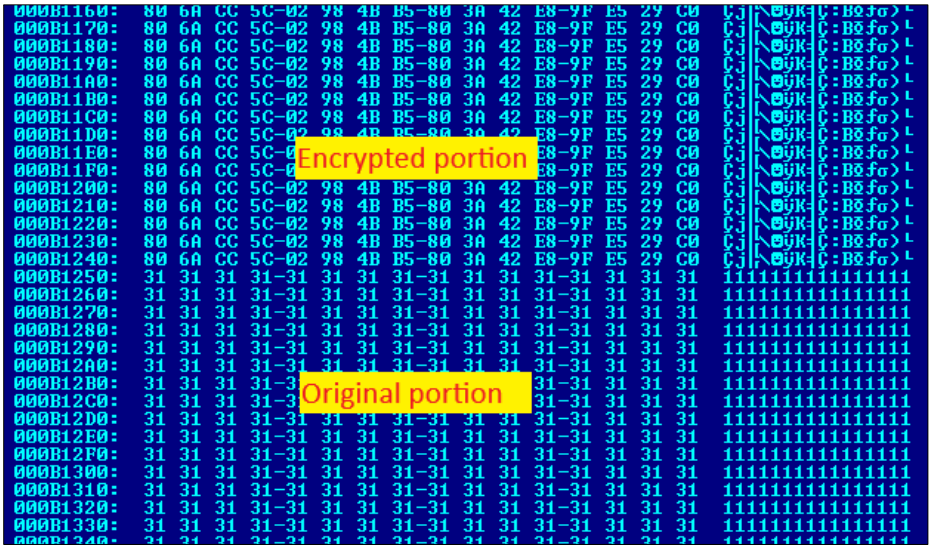


Figure 14: Exemple de cryptage intermittent en mode rapide et lent

Comme le cryptage est assez complexe, nous ne notons que les parties importantes, comme suit :

- Marqueur d'infection - TIMATOMA# ou TIMATOMAFULL# pour un cryptage complet
- ID de décryptage - Les ID peuvent être trouvés dans la note de rançon et le fichier DECRYPTION_IDS.TXT.
- Blocs cryptés - contient les décalages de début et de fin des blocs cryptés.

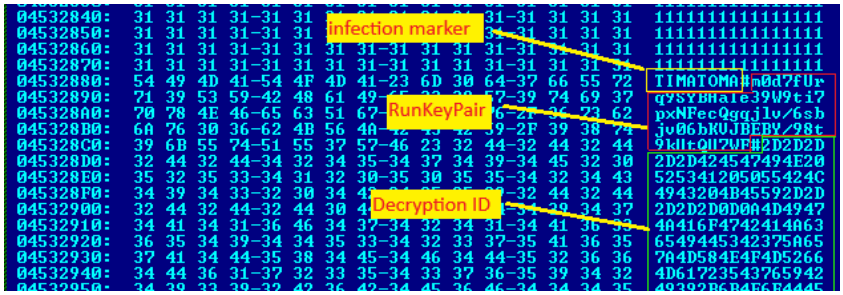


Figure 15: Marqueur d'infection et autres informations stockées à la fin du fichier crypté

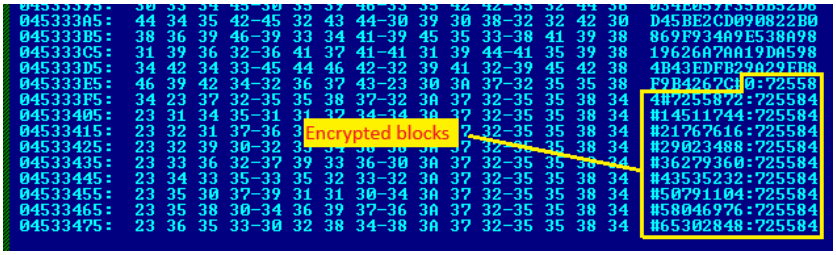


Figure 16: Blocs cryptés commençant par le décalage zéro jusqu'à 725584 en décimal, et ainsi de suite.

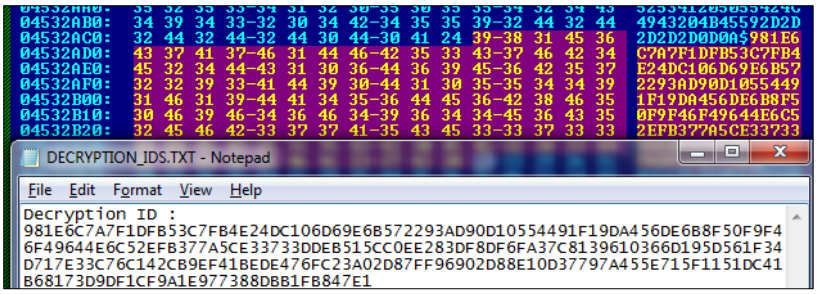


Figure 17: L'ID de décryptage ajouté au fichier crypté se trouve dans le fichier DECRYPTION_IDS.TXT.

Au cours du processus de cryptage, les noms des fichiers cryptés sont encodés en Base64 et complétés par ***.Encrypting** afin d'éviter toute réinfection. Si les fichiers sont déjà cryptés, ils sont complétés par ***.Encrypted**.

```
if ( sub_6F1BB4(v132, v128, v127) )
{
  r_concatenate_string(&v67, v128, L".Encrypted");
  LOBYTE(v30) = 1;
  v118 = sub_477604(&off_44E590, v30, v67, 2, 0x8000);
  v50 = 0LL;
  LOBYTE(v31) = 2;
  (*(*v118 + 40))(v118, v31);
  sub_4732D4(v111, &v65);
  r_concatenate_string(&v66, L"TIMATOMAFULL#", v65);
```

*Figure 18: Le mode de cryptage complet ajoute **TIMATOMAFULL#** comme marqueur, puis renomme le fichier crypté avec l'extension ***.Encrypted**.*

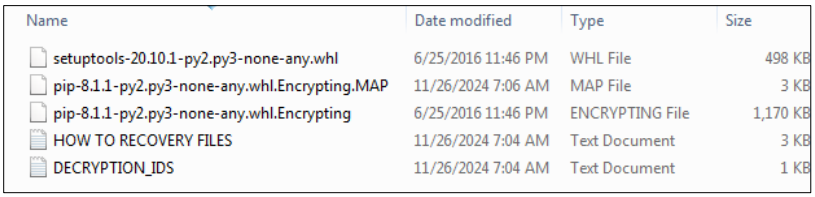


*Figure 19: ajoute l'extension ***.Encrypting** pendant le chiffrement et ajoute ***.Encrypted** une fois le chiffrement terminé, ce qui inclut également l'ajout de la note de rançon "**HOW TO RECOVER FILES.txt**" et du fichier "**DECRYPTION_IDS.txt**"*

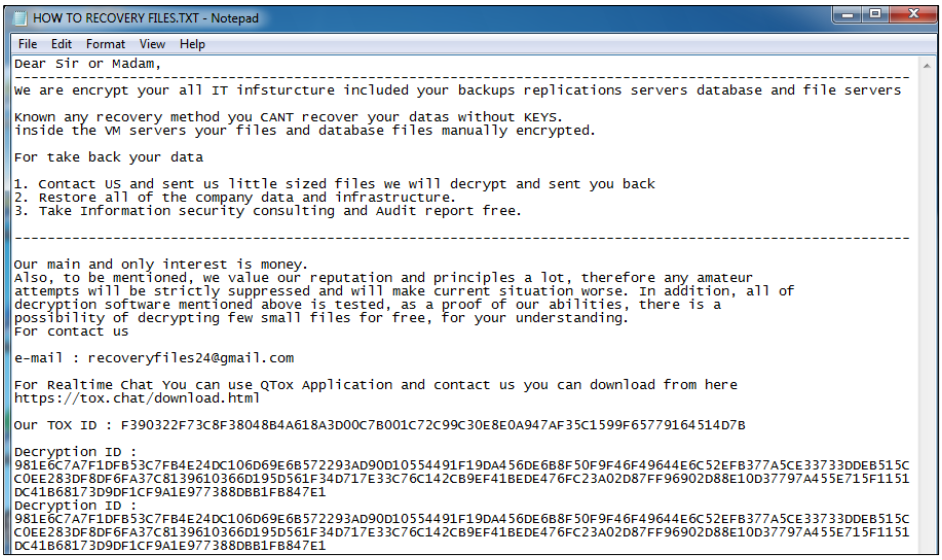


Figure 20: Note de rançon contenant les identifiants de décryptage et l'email de contact

Enfin, si la case "**Fermer après**" est cochée, un fichier batch d'autosuppression sera déposé et exécuté.

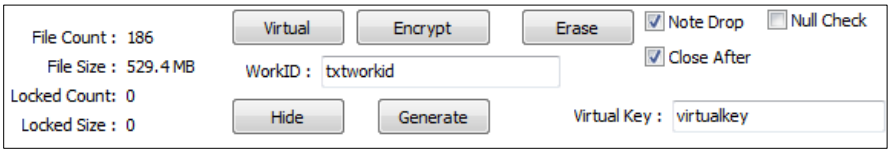


Figure 21: Interface utilisateur avec cases à cocher et autres paramètres

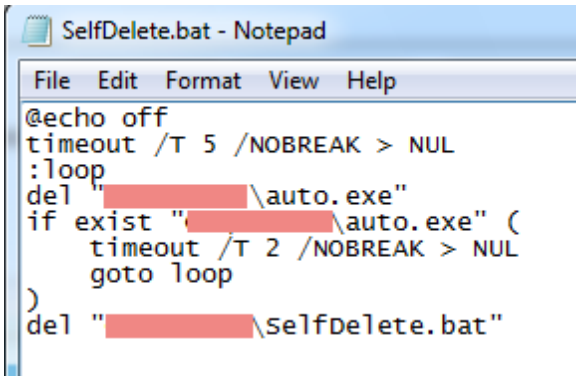
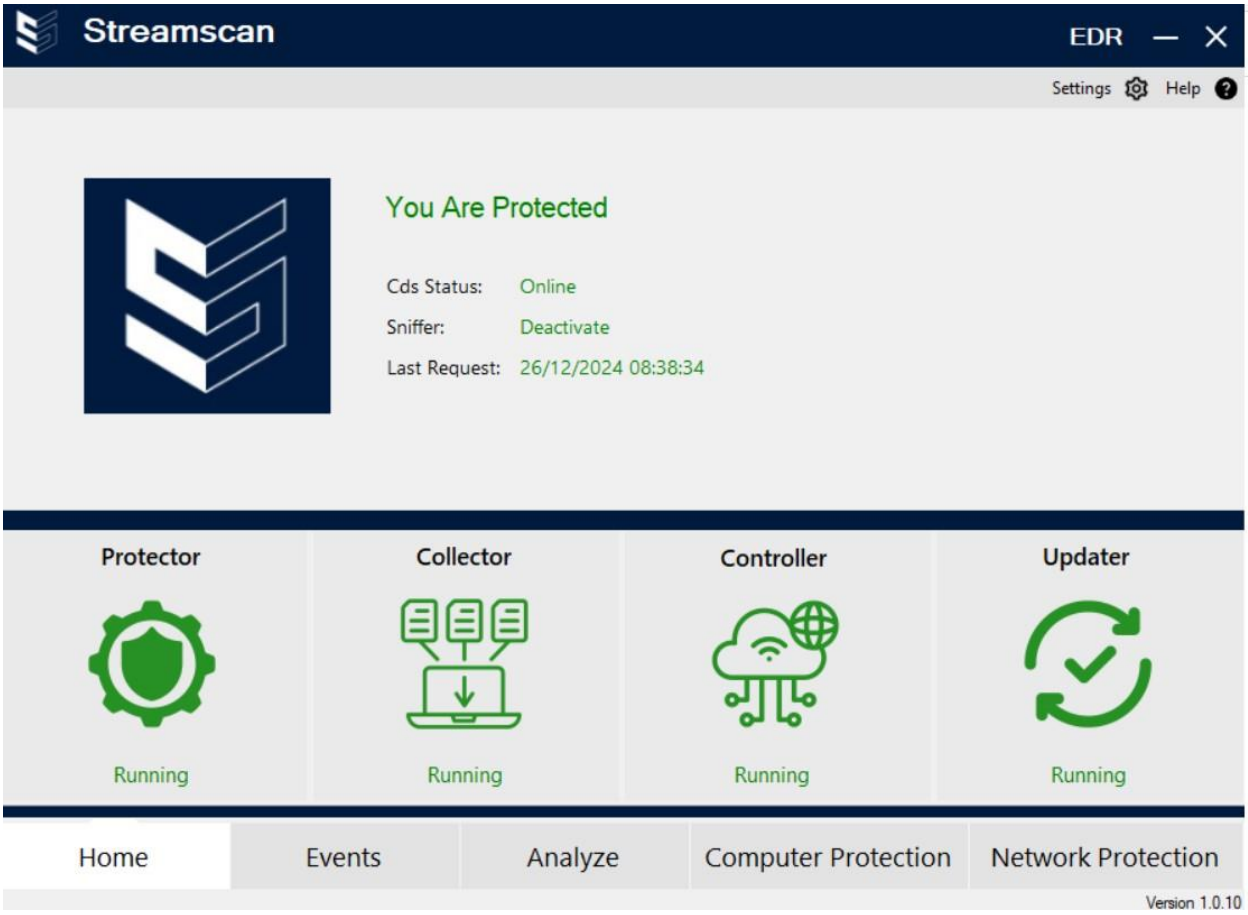
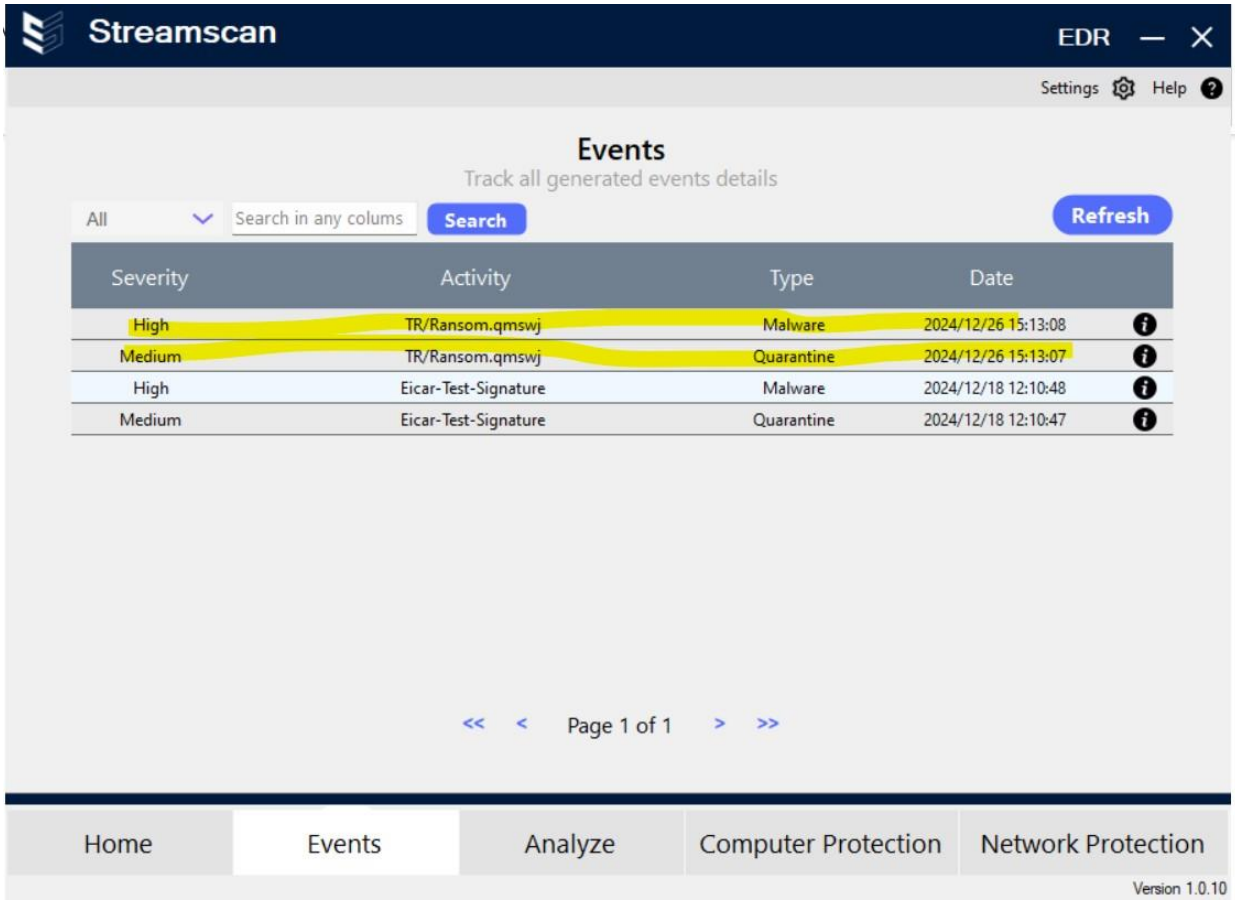


Figure 22: Auto-effacement par dépôt d'un fichier batch dans le même chemin d'accès au fichier auto.exe

Detection par notre Streamscan EDR

Ce rançongiciel est détecté et bloqué par l'EDR de Streamscan.





Conclusion :

Il semble que Space Colon continue d'améliorer son arsenal en ajoutant des fonctions que d'autres n'ont pas encore mises en œuvre, en modifiant la logique de cryptage, mais la complexité peut provoquer des erreurs ou empêcher de récupérer les fichiers cryptés.

Annexe :

Annexe A. Extensions qui sont chiffrés par le ransomware

.mdf;.ldf;*.ndf;*.dbf;*.db;*.dbs;*.log1;*.dat;*.mdb;*.ora;*.fdb;*.wt;*.gdb;*.bak;*.backup;*.vib;*.vbk;*.vbm;*.tib;*.tibx;*.dmp;*.tuf;*.trn;*.zip;*.7z;*.rar;*.archive;*.vhd;*.vhdx;*.avhdx;*.rct;*.pst;*.sql;*.pdf;*.xlsx;*.xls;*.doc;*.docx;*.pstx;*.vmdk;*.vmx;*.fbk;*.ibd;*.myd;*.mdbx;*.d0;*.d1;*.d3;*.b1;*.bck;*.tiff;*.tif;*.dwg;*.dxf;*.lbl;*.db1;*.001;*.0001;*.seq;*.vix;*.bkf;*.gbk;*.ova;*.eml;*.ptb;*.saj;*.sko;*.skp;*.srd;*.nyf;*.itdb;*.dbc;*.edb;*.nsf;*.ib;*.db2;*.ai;*.ost;*.ostx;*.002;*.003;*.004;*.005;*.006;*.007;*.cbd;*.d2;*.d4;*.da1;*.da2;*.da3;*.da4;*.rpd;*.edp;*.vrb;*.vswp;*.idx;*.ebk;*.rpt;*.vct;*.vcx;*.cdx;*.hbp;*.msg;*.ldb;*.alt;*.dbw;*.qrp;*.upd;*.old;*.jet;*.ol2;*.sic;*.bco;*.hrl;*.a06;*.a01;*.a02;*.a03;*.bac;*.3dm;*.3dmbak;*.psd;*.jpg;*.mp4;*.mov;*.pptx;*.ppt;*.tmp;*.cdr;*.ard;*.usr;*.btr;*.qvx;*.008;*.hlp;*.sldrpt;*.SLDLFP;*.SLDPRT;*.SLDDRW;*.rfs;*.llp;*.slp;*.spl;*.one;*.SLDASM;*.gan;*.rpo;*.ntf;*.ndk;*.nbf;*.nx1;*.bik;*.dxt5_2d;*.cgd;*.tga;*.avi;*.pak;*._ms;*.danger;*.tar;*.tar.gz;*.frm;*.myi;*.accdb;*.sqlite;*.sqlite3;*.redo;*.log;*.box;*.ndx;*.nsg;*.fmp12;*.fp7;*.fp5;*.couch;*.qbw;*.qbb;*.qba;*.qbm;*.sai;*.xlsm;*.indd;*.3ds;*.max;*.blend;*.qic;*.sna;*.gho;*.ghs;*.iv2i;*.pbd;*.pqb;*.afi;*.mrimg;*.spf;*.bi4;*.rbf;*.obk;*.oeb;*.rdb;*.bup;*.bkup;*.cfgbak;*.nb7;*.lst;*.cbu;*.bpf;*.ctf;*.bkp;*.fbf;*.fbw;*.ful;*.stm;*.imd;*ibdata;*.vmsd;*.vmsn;*.csv;*.elg;*.fxl;*.mtx;*.vob;*.trc;*.arc;*.bdmp;*.dbdmp;*.df;*.vmxf;*.encvrt;*.psm;*.par;*.dft;*.asm;*.adm;*.rtf;*.sev;*.qsm;*.stp;*.mmo;*.sqlaudit;*.rman;*.arc;*.1*;*.2*;*.3*;*.4*;*.5*;*.6*;*.7*;*.8*;*.9*;*.sna;*.prt;*.tgz;*.wim;*.raw;*.pri;*.pcb;*.mat;*.pod;*.step;*.c1;*.axf;*.sldprt;*.BBCK;*.BBCK3;*.pdb;*.full;*.diff;*.bin;*.zip

Annexe B. Le ransomware efface les extensions suivantes :

.bak ; .backup ; .vib ; .vbk ; .vbm ; .tib ; .tibx ; .vrb ; .bco ; .bucket

Annexe C. Extensions virtuelles - Extensions liées aux machines virtuelles :

.vmdk ; .vhd ; .vhdx ; .avhdx ; .vdi ; .encvrt

Annexe D. Extensions lentes :

.mdf ; .ldf ; .dbf ; .ora ; .nsf ; .ib ; .ibd ; .fdb ; .myd

Annexe E. Fichiers et extensions sur liste noire (non chiffrées)

NTUSER.dat	windows.edb	.msi
icudtl.dat	.exe	.cab
NTUSER.dat.log1	.dll	unins0
usrclass.dat	.ocx	\NFenêtres\N
usrclass.dat.log1	.sys	c:\NFenêtres
webcachev01.dat	.crypté	.bat
pouces.db	.cryptage	.com
Fichiers communs	.encrypting.map	.inf

.ini	.reg	settings.dat
.cmd	.nt	\Fichiers communs
.drv	.ntfs	mpenginedb.db

Les CIO :

Nom de fichier/Autres	SHA1	Description
auto.exe	28d0d07c190fd9827560d3e59dd70a8a26392f82	Space Colon ransomware
%windir%\AdminArsenal\PDQDeployRunner\service-1\PDQDeployRunner-1.exe		Utilisation potentielle du PDQ Deploy pour déployer silencieusement un ransomware, comme on peut le voir dans les journaux d'événements.
procdump.exe	f4d82f11ca773a5606a2bc07add94a0cc76827ec	Possibilité d'abus de ProcDump de SysInternals en vidant la mémoire de processus critiques comme LSASS
SelfDelete.bat		Utilisé pour se supprimer lui-même.
COMMENT RECOUVRIR LES FICHIERS.txt		Note de rançon
DECRYPTION_IDS.txt		ID de décryptage
F390322F73C8F38048B4A618A3D00C7B001C72C99C30E8E0A947AF35C1599F65779164514D7B		ID TOX utilisé par l'acteur de la menace comme canal de communication
recoveryfiles24[@]gmail[.]com		Adresse électronique de l'auteur de la menace

TTPs :

Catégorie	ID	Nom	Description
Développement des ressources	T1588.002	Obtenir des capacités : Outil	L'acteur de la menace installe potentiellement le PDQ Deploy, qui a déjà été utilisé par un autre ransomware.
Exécution	T1204	Exécution par l'utilisateur	Le ransomware repose sur l'exécution par l'utilisateur, car il utilise des outils tiers qui nécessitent une interaction avec l'utilisateur.

			Ex. procdump.exe, PDQ Deploy
	T1059.003	Interprète de commandes et de scripts : Shell de commande Windows	Exécute SelfDelete.bat
Défense Evasion	T1057	Découverte du processus	Énumérer les processus et les services
	T1562.001	Affaiblir les défenses : Désactiver ou modifier des outils	Met fin aux processus ou aux services susceptibles d'empêcher le cryptage des fichiers.
	T1140	Désobfusquer/décoder des fichiers ou des informations	Les chaînes sont codées en Base64 et utilisent des algorithmes de cryptage avancés pour crypter les fichiers.
	T1070.004	Suppression des indicateurs : Suppression de fichiers	Se supprime lui-même à l'aide d'un fichier batch.
Impact	T1485	Destruction des données	Certains fichiers sont irrécupérables en raison de la logique complexe mise en œuvre sur le site , qui est sujette à des erreurs.
	T1486	Des données cryptées pour plus d'impact	Cryptage des fichiers sensibles