VSRC 安全周报（2021-06-08）

0x00 本周漏洞综述

本周需要关注漏洞共 2 个：西门子 PLC 远程代码执行漏洞（CVE-2020-15782）；Cisco 6 月多个安全漏洞。

本周安全态势共 1 个：Kaspersky：JSWorm 勒索软件演变分析。

根据以上综述，本周安全威胁为中。

0x01 重要安全漏洞列表

1. 西门子 PLC 远程代码执行漏洞（CVE-2020-15782）

PLC(可编程逻辑控制器)是一种专门为工业环境应用而设计的数字运算操作电子系统。它采用一种可编程的存储器，在其内部存储执行逻辑运算、顺序控制、定时、计数和算术运算等操作的指令，通过数字式或模拟式的输入输出来控制各种类型的机械设备或生产过程。

2021 年 05 月 28 日，Claroty 的研究人员公开披露了 Siemens（西门子）PLC 中的一个远程代码执行漏洞（CVE-2020-15782），其 CVSS 评分为 8.1。能够网络访问 TCP 端口 102 的远程攻击者可以利用该漏洞绕过 PLC CPU 中的 PLC 沙箱，在受保护的内存区域中写入或读取数据，最终远程执行恶意代码，且该漏洞无需经过身份验证即可利用。

攻击者可以在禁用访问保护的 PLC 上滥用此漏洞，以获得 PLC 上任何位置的读写访问权限并远程执行恶意代码，并且利用此漏洞的攻击将很难被检测。

影响范围

Affected Product and Versions	Remediation
SIMATIC Drive Controller family: All versions < V2.9.2	Update to V2.9.2 or later version https://support.industry.siemens.com/cs/ww/en/view/109773914/
SIMATIC ET 200SP Open Controller CPU 1515SP PC2 (incl. SIPLUS variants): All versions	See recommendations from section Workarounds and Mitigations
SIMATIC ET 200SP Open Controller CPU 1515SP PC (incl. SIPLUS variants): All versions	See recommendations from section Workarounds and Mitigations
SIMATIC S7-1200 CPU family (incl. SIPLUS variants): All versions < V4.5.0	Update to V4.5.0 or later version https://support.industry.siemens.com/cs/ww/en/view/109793280/
SIMATIC S7-1500 CPU family (incl. related ET200 CPUs and SIPLUS variants): All versions < V2.9.2	Update to V2.9.2 or later version https://support.industry.siemens.com/cs/ww/en/view/109478459/
SIMATIC S7-1500 Software Controller: All versions	See recommendations from section Workarounds and Mitigations
SIMATIC S7-PLCSIM Advanced: All versions < V4.0	Update to V4.0 or later version https://support.industry.siemens.com/cs/ww/en/view/109795016/

安全建议

目前 Siemens 已经修复了此漏洞，建议参考官方发布的安全咨询及时升级更新:

下载链接：

https://cert-portal.siemens.com/productcert/pdf/ssa-434534.pdf

参考连接：

https://cert-portal.siemens.com/productcert/pdf/ssa-434534.pdf

https://claroty.com/2021/05/28/blog-research-race-to-native-code-execution-in-plcs/

https://securityaffairs.co/wordpress/118367/ics-scada/cve-2020-15782-siemens-plcs-flaw.html?

https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-15782

2. Cisco 6月多个安全漏洞

2021年06月02日，Cisco发布安全公告，修复了包括Webex Player、SD-WAN 软件和ASR 5000 系列软件中的多个安全漏洞，攻击者可以通过利用这些漏洞提升权限或在受影响的系统上执行任意代码。

在本次修复的高危漏洞中,CVE-2021-1503、CVE-2021-1526和CVE-2021-1502都是Cisco Webex中的内存损坏漏洞，CVSS评分均为7.8。由于对高级录制格式（ARF）或 Webex 录制格式（WRF）的 Webex 录制文件中的值验证不足，攻击者可以通过链接或电子邮件附件向用户发送恶意 ARF 或 WRF 文件并诱导用户打开该文件来利用这些漏洞，最终导致攻击者使用目标用户的权限在受影响的系统上执行任意代码。

CVE-2021-1528是Cisco SD-WAN 软件CLI 中的一个提权漏洞，CVSS评分为7.8，由于受影响的软件没有正确限制对特权进程的访问,经过身份验证的本地攻击者可以通过调用受影响系统中的特权进程来利用此漏洞，最终能够使用root用户的权限执行操作。

CVE-2021-1539和CVE-2021-1540是Cisco ASR 5000 系列软件（StarOS）授权过程中的漏洞，CVSS评分分别为8.1和6.5。由于非交互式 CLI 命令的错误授权，攻击者可以通过向受影响的设备发送恶意 SSH 请求来利用此漏洞，最终经过身份验证的远程攻击者能够绕过 TACACS 授权或nocli 授权，并在受影响的设备上执行 CLI 命令。

CVE-ID	类型	影响	影响范围
CVE-2021-1502	验证不足、内存损坏	任意代码执行	Windows 和 macOS 版： Cisco Webex Network Recording Player 及41.4版本之前的 Cisco Webex Player
CVE-2021-1503			Windows 和 macOS 版： Cisco Webex Network Recording Player 及41.2版本之前的 Cisco Webex Player
CVE-2021-1526			Windows 和 MacOS 版：41.5版本之前的 Cisco Webex

			Player
CVE-2021-1528	访问限制不当	权限提升	运行 Cisco SD-WAN 软件版本 20.4、20.5 的以下产品： SD-WAN vBond Orchestrator Software SD-WAN vEdge Cloud Routers SD-WAN vEdge Routers SD-WAN vManage Software SD-WAN vSmart Controller Software
CVE-2021-1539	授权错误	TACACS 授权绕过	运行 Cisco StarOS 版本（21.16 之前版本、21.16、21.17、21.18、21.19、21.19.n、21.20）的以下 Cisco 产品： ASR 5000 Series Aggregation Services Routers Virtualized Packet Core - Distributed Instance (VPC-DI) Virtualized Packet Core - Single Instance (VPC-SI)
CVE-2021-1540		nocli 授权绕过	

安全建议

目前 Cisco 已经修复了这些漏洞，建议参考官方安全公告及时升级更新：

参考连接：

https://tools.cisco.com/security/center/publicationListing.x

参考连接：

https://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-asr5k-autho-bypass-mJDF5S7n

https://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-webex-player-kOf8zVT

https://securityaffairs.co/wordpress/118564/security/cisco-webex-player-sd-wan-asr-5000-flaws.html?

0x02 本周安全态势

1. Kaspersky：JSWorm 勒索软件演变分析

介绍

在过去的几年里，勒索软件的威胁形势一直在发生变化，我们见证了一种模式的转变：从 2017 年勒索软件大规模爆发（如 WannaCry、NotPetya 和 Bad Rabbit）开始，很多勒索软件背后的攻击者已经转向隐蔽且高利润的 big-game hunting（大型游戏狩猎，即针对知名度高的组织和机构）战术，并且勒索软件导致全球某些公司服务中断已经变得司空见惯。

在某些情况下，这种全球趋势只是反映了威胁的持续生命周期：旧的勒索软件家族关闭，新的勒索软件出现并追求新的目标。但是，在近两年里，某些单一的勒索软件家族已经从一个大规模的攻击活动演变为一个具有高度针对性的威胁。在本文中，我们分析了一个名为 JSWorm 的勒索软件的发展史。

年表

JSWorm 勒索软件在 2019 年被发现，从那时起，不同的变体被命名为不同的名字，如 Nemty、Nefilim、Offwhite 和其它几个。

每个更名变体的一部分都发布了多个版本，这些变体更改了代码、文件扩展名、加密方式和加密密钥等不同方面。

在下图中，我们介绍了该恶意软件使用的一些名称，以及相应的变体在野外（ITW）积极分发的日期（而不是首次发现的日期），虽然该列表并不全面，但是它标志着 JSWorm 演变过程中的主要里程碑。

除了名称更改外，该勒索软件的开发人员还一直在重新设计其代码，并尝试使用不同的分发方法。

2020 年某个阶段时，开发人员甚至将编程语言从 C ++更改为 Golang，从头开始完全重写代码。但是，加密方式、赎金记录的相似性和使用相同数据泄漏网站地址使我们确信这是

相同的攻击活动。

该恶意软件的原始版本以及之后的一些“更名”，例如 Nemty，是由一个用户名为 jsworm 的发帖人在暗网上宣传的。

I gonna present you new Ransomware - JSWORM.
Functions:

- Native C++.
- No passes.
- Encrypting just a part of file.
- Ransom note in each encrypted folder (exception Windows)
- Multy-threaded (for each disk there is one thread)
- Delete shadow copies, stop services (sql and etc), kill tasks (sql and etc), clear logs.
- Self-delete after all tasks.
- Random 256 symbols password.

What you need:

- Traffic (rdp, vnc, push, botnet, spam, exploits).
- Two abuze e-mails (I can advice you cock.li or tutanota).

Conditions:

- I give you build with your e-mails.
- The first income must be within 2 weeks.
- Work 70/30, 70% yours, 30% mine.
- Income pay to your cryptowallet.
- Decryptor after my 30%.

Contacts:

- Jabber: jsworm@exploit.im

早期 JSWorm 变体的宣传

分发方式

从 2019 年创建到 2020 年上半年，JSWorm 被作为公共 RaaS 提供，并通过以下方式传播：

- RIG 漏洞利用工具包
- Trik 僵尸网络
- 虚假付款网站
- 垃圾邮件活动

从 2020 年上半年开始，公共 RaaS 被关闭，背后的攻击者转而从事大型游戏狩猎。有证据表明，初始访问利用了脆弱的服务器端软件（Citrix ADC）和不安全的 RDP。

技术细节

我们并没有分析这个恶意软件发现的所有变体，因为它们太多，因此我们分析了 SWorm 家族的一些著名变体。其中，日期表示相应变体被在野发现的大致时间。

2019 年 5 月：JSWorm

该样本是 JSWorm 勒索软件最早发现的变体之一，与它的后续软件不同，它不包含内部版本号。该样本使用 C ++开发，并在 MS Visual Studio 中进行编译。

除文件加密外，它还执行诸如停止大量正在运行的进程和服务之类的操作，以最大限度的增加可用于加密的文件数量。此外，它删除所有系统备份和卷影副本，禁用系统恢复模式并清除事件日志。

加密方式

使用带有 256 位密钥的 Blowfish 密码的自定义修改对文件进行加密。密钥是在程序执行开始时根据字符串的连接生成的：用户名、设备 MAC 地址和卷序列号（详见注释中的示例值）。

```
GetUserNameA(Buffer, &buffer_size);           // Username: user
std::string::string(username, Buffer);
GetMacAddress(mac);                           // Mac: D0:00:CO:FE:EE:EE
_mac = v0;
VolumeSerial = GetVolumeSerial(v14);          // Volume Serial: 3141592653
buffer = std::string::append(VolumeSerial, v15, _mac);// 3141592653D0:00:CO:FE:EE:EE
std::string::append(buffer, v19, username);   //         3141592653D0:00:CO:FE:EE:EEuser
```

密钥生成过程

然后，生成由赎金票据称为“ JSWORM PUBLIC KEY”的字符串。事实上，勒索软件开发者所谓的“ JSWORM PUBLIC KEY”是上述的 Blowfish 密钥与字符串“ KCQKCQKCQKCQ”进行 XOR 运算，并以 Base64 编码生成的，并没有使用非对称加密技术，在这种情况下，使用“public”一词是没有意义的。

```
qmemcpy(key, "KCQKCQKCQKCQ", sizeof(key));
i = 0;
v6[4] = 0;
v6[5] = 0;
sub_7D4F64(v6, &volume_mac_username);
if ( size )
{
  do
  {
    p_volume_mac_username = &volume_mac_username;
    _blowfish_key = blowfish_key;
    if ( a6 >= 0x10 )
      p_volume_mac_username = volume_mac_username;
    if ( blowfish_key[5] >= 0x10u )
      _blowfish_key = *blowfish_key;
    *(_blowfish_key + i) = p_volume_mac_username[i] ^ key[i % 12];
    ++i;
  }
  while ( i < size );
}
```

与“ KCQKCQKCQKCQ”进行 XOR

下面是一个密钥计算的示例，其中选择了序列号和 MAC 地址：

```
Blowfish key: “53385773534FE:ED:00:DE:AF:00user”
XOR 之后的公钥: “5xpi^tfxvb\x05\x14q\x06\x15qsaq\x07\x14q\x02\x17qsa>049”
转换为 Base64 后的公钥: “NXhwaX50Znh2Yn8FFHEGFXFzYXEHFHECF3FzYT4wNDk=”
```

自定义版本的 Blowfish 被用来对每个受害者的文件内容进行加密，最多可以加密 10 万个字节，这可能是为了加快大文件的加密速度，加密的数据将覆盖原始数据。

开发人员更改了 Blowfish 密码的内部实现，这导致它与标准实现不兼容，也可能是为了使研究人员更难解密。

在对文件内容进行加密后，该程序会将其重命名。一个附加扩展名 ".[ID-...][mail@domain.tld].JSWORM "将被添加到文件名中。

加密缺陷

该恶意软件基本上是将可用于解密的密钥保存在赎金票据中。对其进行 Base64 解码和解密是微不足道的，无需支付赎金就可以保存受害者的数据，即使由于某种原因丢失了赎金票据，也可以在受感染的机器上轻易地重新生成密钥。

2019 年 7 月：JSWorm 4.0.3

MD5：5444336139b1b9df54e390b73349a168

JSWorm 的改进和更新版本，试图修复以前版本中发现的缺陷。

该样本包含对受感染机器的语言检查。这很可能是为了防止在使用以下语言的系统上对数据进行加密：RU（俄语），BE（白俄罗斯语），UZ（乌兹别克语），KY（吉尔吉斯语），TG（塔吉克语），TK（土库曼语），KK（哈萨克语），UK（乌克兰语）。

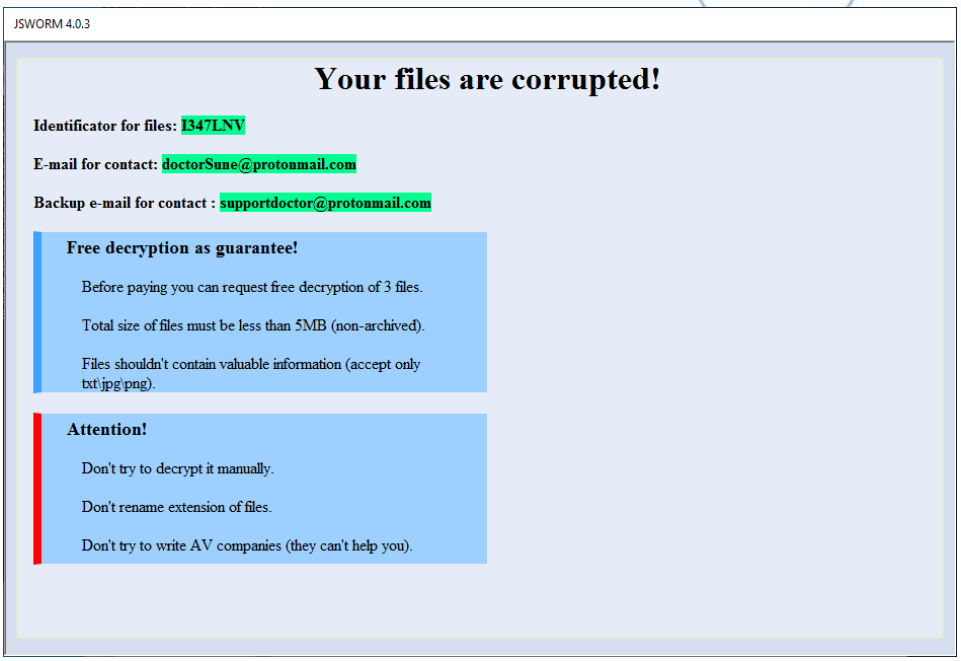
```
if ( (GetUserDefaultLangID() & 0x3FF) != LANG_RUSSIAN
  || (GetUserDefaultLangID() & 0x3FF) == LANG_BELARUSIAN
  || (GetUserDefaultLangID() & 0x3FF) == LANG_UZBEK
  || (GetUserDefaultLangID() & 0x3FF) == LANG_KYRGYZ
  || (GetUserDefaultLangID() & 0x3FF) == LANG_TAJIK
  || (GetUserDefaultLangID() & 0x3FF) == LANG_TURKMEN
  || (GetUserDefaultLangID() & 0x3FF) == LANG_KAZAK
  || (GetUserDefaultLangID() & 0x3FF) == LANG_UKRAINIAN )
{
```

确定用户的语言

然而，可能是错误导致，这个版本的勒索软件只在系统语言为俄语的情况下才会加密文件。如果我们仔细观察上面的条件，我们可以看到第一个条件是'！='（不等于），这使得木马在语言不是俄语的系统上没有加密即退出。如果条件是'=='，就会采用另一个分支，其结果可能是该木马最初想要的。

此变体中的赎金记录是一个名为<ID> -DECRYPT.hta 的 HTA 文件，其中<ID>是恶意软件分配的唯一受害者 ID，该 HTA 文件在文件加密完成后启动，并通过注册表添加到自动运行中：

```
reg add HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run /v “zapiska” /d
“C:\Users\user\JSWRM-DECRYPT.hta”
reg add HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run /v “zapiska” /d
“C:\Users\user\JSWRM-DECRYPT.hta”
```



JSWorm 4.0.3 的赎金说明

加密方式

这个版本的 JSWorm 使用 RSA 的 WinAPI 实现和 AES 的自定义实现来加密文件。JSWorm 生成两个 128 位（IV）和 256 位（密钥）的随机值，这些随机值仅限于字符：a...z，A...Z，和 0...9。RSA 公钥嵌入在勒索软件内部：

```
db 'BgIAAACkAABSU0ExABAAAAEAAQBdNHPJdGGanQOPD9jYrShdFuf2Dl7swZcYPWIGn'
                        ; DATA XREF: .text:00591DA8↑o
                        ; RsaEncrypt+29↓o ...
db 'YGBNFJgBkjMCkRNUybsmgiTCtF9PJRmPQCtY7ro3EayvLZTmPsCD0oKlrEvH7v3a5'
db 'OIaixe1zqm61rRpupfcoL6F7AA9biaHBH+NYMf7ymhvr1gtQeV5C1Sh1sYDHWuI+t'
db 'WOeeB8I4jHgv5pECe7/YhkwRaSuurzOZFoB+SaRieFtL5t7iVOr7q7i668cXoR/0S'
db '9RnJAk9jM8+aKyTs4/QQdRSwdOtZ3t7JOVFH9Bk2USBSuHKpZKmNKrJa2vciAMZtz'
db 'U7OGKSxWygcVdY8/bu08hSorqh+OJNvHVvVIJRBU1lInSz7vDTJCb0AsVCMBG5o2R'
db 'atsuMdR17fdySArLzdmFFYFELx1FNv8RvfKNgNAUUIBiAJxH8T4vRaZ7bTr9dJiBy'
db 'l0pnJwa5GQqtKRl8o+nynYfQzZjDdaG5cBjls1aPa/ltkinTHAvDwDCz5YJSpr9mH'
db '89/fmhOzsptE5dCNxwuh6QuAGF74Lg4hhf2rA6RJFvOLRruCaNjnLmmCzjY47iaXK'
db 'P6C0z58Ol3jRU6WQ+tTPaPVgkpjdh739p8RdlMZaPIeHe62131pNBfq9YIGSUBnGi'
db 'fqyK2YNHvwC3S7j3uFDOQkPt7Si4t1VfjTNAFiQ/KSOSrK/TEqGQqPt8xvtw==',0
align 4
```

JSWorm 4.0.3 中使用的 RSA 公钥

使用这个密钥，JSWorm 加密 AES 密钥和初始化向量（IV）并将其 Base64 编码：

```
if ( !CryptStringToBinaryA(pszString, 0x2C8u, CRYPT_STRING_BASE64, 0, &pcbBinary, 0, 0) )
  ExitThread(0);
pbBinary = (BYTE *)malloc(pcbBinary);
pbData = pbBinary;
if ( !pbBinary )
  ExitThread(0);
if ( !CryptStringToBinaryA(pszString, 0x2C8u, CRYPT_STRING_BASE64, pbBinary, &pcbBinary, 0, 0) )
  ExitThread(0);
if ( !CryptAcquireContextA(&phProv, 0, szProvider, PROV_RSA_FULL, 0)
  && !CryptAcquireContextA(&phProv, 0, szProvider, PROV_RSA_FULL, CRYPT_NEWKEYSET) )
{
  ExitThread(0);
}
if ( !CryptImportKey(phProv, pbData, pcbBinary, 0, 0, &phKey) )
  ExitThread(0);
pdwDataLen = 0;
v3 = 49;
if ( !CryptEncrypt(phKey, 0, 1, 0, 0, &pdwDataLen, 0) )
  ExitThread(0);
if ( !CryptEncrypt(phKey, 0, 1, 0, g_rand32_and_16, &v3, pdwDataLen) )
  ExitThread(0);
pcchString = 0;
if ( !CryptBinaryToStringA(g_rand32_and_16, pdwDataLen, CRYPT_STRING_BASE64, 0, &pcchString) )
  ExitThread(0);
v1 = (CHAR *)operator new[](pcchString);
if ( !CryptBinaryToStringA(g_rand32_and_16, pdwDataLen, CRYPT_STRING_BASE64, v1, &pcchString) )
  ExitThread(0);
```

JSWorm 4.0.3 中的 WinAPI RSA 加密

之后，这个值会被添加到勒索软件说明<ID>-DECRYPT.hta 中，但这个值不会被显示出来，因为它位于文件内部，是一个 HTML 注释。

```
<html><head><title>JSWORM 4.0.3</title><HTA:APPLICATION APPLICATIONNAME=
padding-left: 10px;"><h3 style="padding-left: 15px; padding-top: 5px;">
<!--
6cxofacKhoDAITACTIZSR4+BWDAPpOVuJnwr4YgcPFpIGh8z36T0NPgjIPq2t+23
ff/Zd9SnoC/V54soL91WTXSqJ0Ego889T86MIbbGDAiunXKTXZHTQzF5CMM934yU
EPIUR2xx1wd7naGbRinC3l8JVw4osbsYLabiP/Q+AsKcwO3Wk5x3W8LHqCRzTbRx
Qow75iDu0+npCapronV1ne7BzVe5tPvpzMb2Vjj8ChzLKJYKPou+dTGJdW5/SoKV
x+TWhTXq9Ygk9Cqadq+FSFXFqpG1zPc/ugS0elzmmgKuztSI9aLdunBpsDxfSRa3
FGwuWS5u0Vi9mulSpwgkkiU7DJxaNQYiOcqv7vNHreBJdciQaJ1L0gsw5mRmg65I
M403NK3oPmJhN8Ui5hpWlsg3L+OBokE41ZAR8VmCW8Um51/ZvIoFK3/UBJUfhy5v
LJCjnm8g5N38ugu35Wlxd2GBhvA6MKWTKnw1M2VBdwOa1A3VgSeEC66FRcM6JnOc
sfce5DhgJwXROQtQboQ8ZzyGNTkgIjDfDMTU9mT28idyP0D5jNmMJ+b+dyBUHK3c
cz3WYhMPeQ9d5ZNyzhzQTQmwZ67FMZU/fNy00+xDWKhlqKKNj8su4+d9oUXX+mTs
FQr12E3V2Cm9SSe2OY5vk5Ia/bOEI6JgcpI24Mmt8T8=
-->
```

.hta 赎金记录文件中的值

为了使研究人员更难解密，恶意软件开发者实施了与标准算法不兼容的 AES 分组密码的自定义变体，使用上述密钥和 IV，通过此密码对受害者文件的内容进行加密。

像以前一样，为了优化，大文件中只有前 16 万字节被加密。加密后，文件名将附加一个扩展名，这一点我们在之前的样本中很熟悉：<filename>.[ID-NQH1J49][doctorSune@protonmail.com].JSWRM”。

加密缺陷

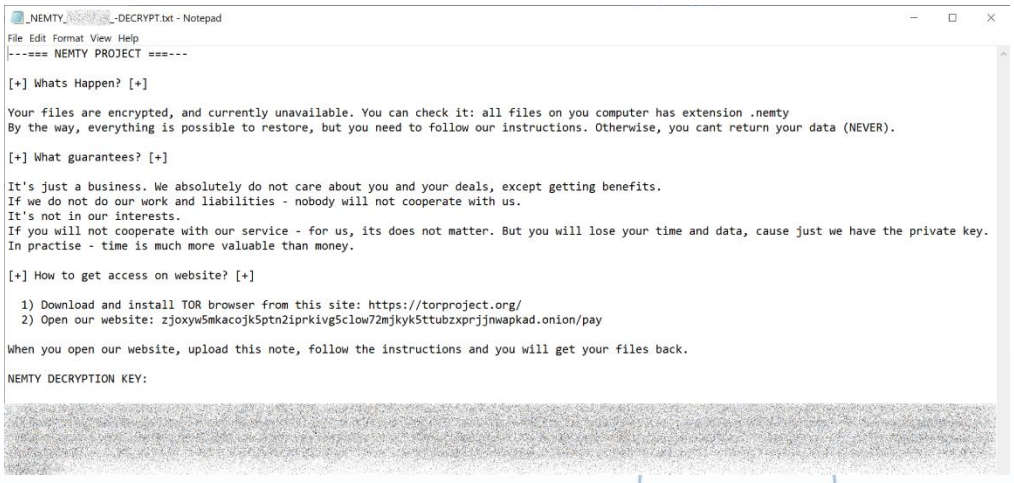
在 JSWorm 的这个变体中，开发者试图修复研究人员在以前版本中发现的缺陷。但是，仍然可以实现不付费解密。用于生成密钥和 IV 的伪随机数生成器在密码学上并不安全，它允许研究人员通过攻击生成算法恢复密钥和 IV，而知道了这些值，就可以解密受害者的数据。

2019 年 8 月：Nemty 1.4

MD5：1780f3a86beceb242aa81afecf6d1c01

JSWorm 和 Nemty 之间的代码变化很大。根据我们的分析，恶意软件开发者可能从头开始重写了他们的木马程序，可能是为了防止研究人员的解密尝试，因为这些解密尝试使 JSWorm 的几个早期变体的受害者能够不付费地恢复他们的数据。

这个样本也是用 C++开发的，在 MS Visual Studio 中编译它实现了一个小的反分析技巧，包括一个字符串混淆算法。字符串（例如，赎金票据名称和内容、RSA 公钥、付款 URL 等）由 RC4 流密码使用硬编码密钥“ fuckav ”加密，并在 Base64 中进行编码。



Nemty 1.4 的赎金说明

启动后，样本将收集连接到受感染机器的存储设备的信息，通过 HTTP 请求 http://api.ipify.org，获得其外部 IP 地址，通过从 http://api.db-ip.com/v2/free/，确定受害者的国家，生成一对 RSA-2048 会话加密密钥，并将所有收集到的信息组合到一个 JSON 结构中。然后，这个结构被攻击者的公共 RSA 密钥加密，并作为 "NEMTY DECRYPTION

KEY ″保存在赎金记录的末尾。


```
{
        "General": {
                "IP": "[redacted]",
                "Country": "[redacted]",
                "ComputerName": "DESKTOP-[redacted]",
                "Username": "[redacted]",
                "OS": "Windows 10",
                "isRU": false,
                "version": "1.4",
                "CompID": "{[redacted]}",
                "FileID": "_NEMTY_[redacted]_",
                "UserID": "5d66999133e3e27b60dca156",
                "key": "Qb3LfIjHvAQdQZPZrJ7Cj1LHFX9AnUkh",
                "pr_key": "BwIAAACkAABSU0EyAAgAAAEAAQB5I+0luDLSXPqSSC
        },
        "Disks": {
                "C:\\": {
                        "DriveType": "FIXED",
                        "TotalSize": "29GB",
                        "UsedSize": "28GB",
                        "FreeSize": "1GB"
                }
        }
}
```


在通过 RSA 加密之前收集的信息

这个结构中的一些感兴趣的选项：

- isRU：指定由其外部 IP 地址确定的受害者国家是否为以下国家之一：俄罗斯、白俄罗斯、哈萨克斯坦、塔吉克斯坦、乌克兰。
- version：木马程序的内部版本。
- CompID：被感染机器的唯一 ID。
- FileID：每次恶意软件启动时产生的感染 ID。
- UserID：联盟的 ID，在木马样本中硬编码。
- key：用于文件加密的 base64 编码的密钥（将在下文讲述）。
- pr_key：base64 编码的私有 RSA-2048 会话密钥（将在下文讲述）。

加密方式

该木马样本包含攻击者硬编码的 RSA-8192 公钥，我们将其称为主 RSA 公钥。

当在受害者的机器上执行时，该木马还生成了一对 RSA-2048 会话密钥，其私钥为上面提到的 pr_key。除此之外，它还生成了一个 256 位的密钥，该密钥将与基于 AES 的自定义

分组密码一起使用。

256 位密钥和 pr_key 由主 RSA 公钥加密并保存在赎金记录中。

在加密每个受害者的文件时，Nemty 1.4 将生成一个 128 位的 IV，并使用带有该 IV 的 256 位密钥通过基于 AES 的自定义密码对文件内容进行加密。IV 由会话公共 RSA 密钥加密，并附加到加密的文件中。

每个加密文件都被重命名，以附加扩展名“ ._NEMTY_ <…> _”，其中跳过的部分是上述感染 ID，即 FileID。

加密缺陷

像 JSWorm 的某些早期变体一样，Nemty 1.4 中加密方案的实现也不是完美无缺的，可以通过两个缺陷对受害者的文件进行解密：

1.用于生成密钥的 PRNG 是不安全的。

2.RSA 会话密钥没有从系统存储中删除。

通过利用第一个缺陷，可以恢复 256 位密钥，而 pr_key 可以利用第二个缺陷来恢复。一旦知道 pr_key，就可以解密 IV，然后使用 256 位密钥和 IV 解密受害者的文件内容。

C&C 通信

该样本从 https://dist.torproject.org/torbrowser/8.5.4/tor-win32-0.4.0.5.zip 下载 TOR 客户端，解压后在受感染的机器上启动。在等待 30 秒后（显然恶意软件开发者认为足够长的时间才能连接到 TOR 网络），木马将有关感染的信息发送到样本中硬编码的 C&C 服务器。

```
zjoxyw5mkacojk5ptn2iprkivg5clow72mjkyk5ttubzxprjjnwapkad.onion
Nemty 1.4 uses HTTP GET with URI /public/gate?data=
```

发送到服务器的信息与每个赎金记录中保存的信息相同，本质上是上述 JSON 结构的加密版本。

Nemty 的其它版本

“Nemty”这个名称一直使用到 2020 年 3 月。最后的一个变体的内部版本为 3.1。

在它创建后的几个月里，发现了几个中间版本的 Nemty。这些变化包括不同的 mutex 名称和 C&C 地址，增加了终止运行进程、停止服务和删除卷影副本的功能，改进了防止无偿解密的加密技术，更改了赎金说明和一些其它调整。

附带说明，这个恶意软件的开发者倾向于用俄语（音译）硬编码字符串，似乎是在开玩笑，也可能是为了获得研究人员的关注。其中一些字符串包含不雅之词，其原为说唱歌曲的引文。

Nemty 2.4 样本中的字符串

Nemty 2.6 样本中的字符串

2020 年 3 月：Nefilim

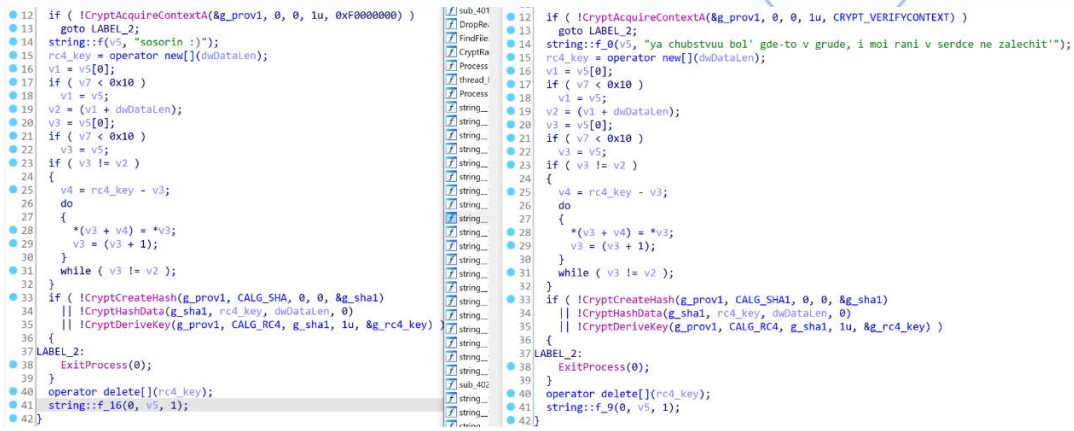
MD5：5ff20e2b723edb2d0fb27df4fc2c4468

大约在 2020 年 3 月，开发人员将其木马的名称更改为 Nefilim。在 Nefilim 的第一个变体出现的时候，这个家族的分发模式发生了变化。开发者从与 JSWorm 和 Nemty 变体一起使用的公共 RaaS 方式转变为与旨在大型游戏狩猎的分支机构的私人合作。其背后的攻击者开始针对备受瞩目的目标，并在受害者的网络内部进行人工操作、泄露机密数据，并威胁恐吓受害者。

木马程序的代码中删除了所有辅助功能，例如进程终止、删除卷影副本、与 C&C 进行通信。该木马程序成为专用于文件加密的单用途二进制文件。如果认为有必要采取任何其它措施，则由攻击者手动执行或在第三方工具的帮助下执行。

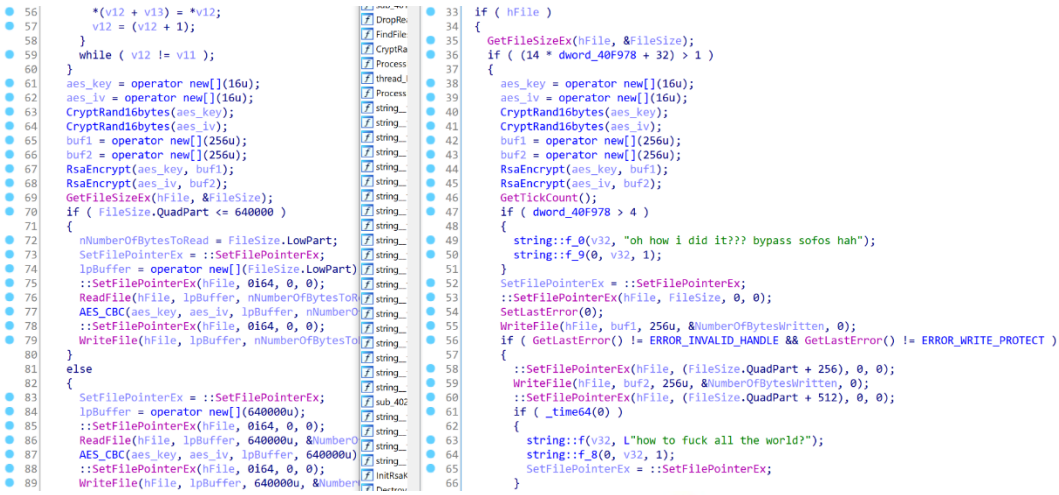
Nefilim 是用 C++开发的，和 Nemty 一样在 MS Visual Studio 中编译，Nemty 的后期版本（2+）和 Nefilim 的代码重合度非常高，这使我们认为是在同一源代码中开发的。

代码重叠的一个例子是相同的字符串解密程序，RC4 密钥是唯一的区别。



（相同的字符串解密程序）左：Nemty 2.6（141dbb1ff0368bd0359972fb5849832d）；右：Nefilim

重叠不仅限于字符串混淆，各种程序的代码片段在整个样本中都是匹配的，包括密钥生成和文件加密功能。



（文件加密过程中的代码相似性）左：Nemty 2.6（141dbb1ff0368bd0359972fb5849832d）；右：Nefilim

与 Nemty 不同，Nefilim 样本是由数字签名签署的，该签名在这个恶意软件变体被积极传播时是有效的，但后来被撤销了。

启动后，该恶意软件会检查命令行参数。如果没有参数，它将继续在所有本地和远程驱动器上搜索受害者的文件；如果提供了参数，则木马会检查它是否是一个现有的目录路径，如果是的话，它将对这个目录中的文件进行加密。否则，它将把该参数解释为一个文件路径，并试图对该文件进行加密。添加命令行参数检查可能是为了让网络犯罪分子手动选择文件进

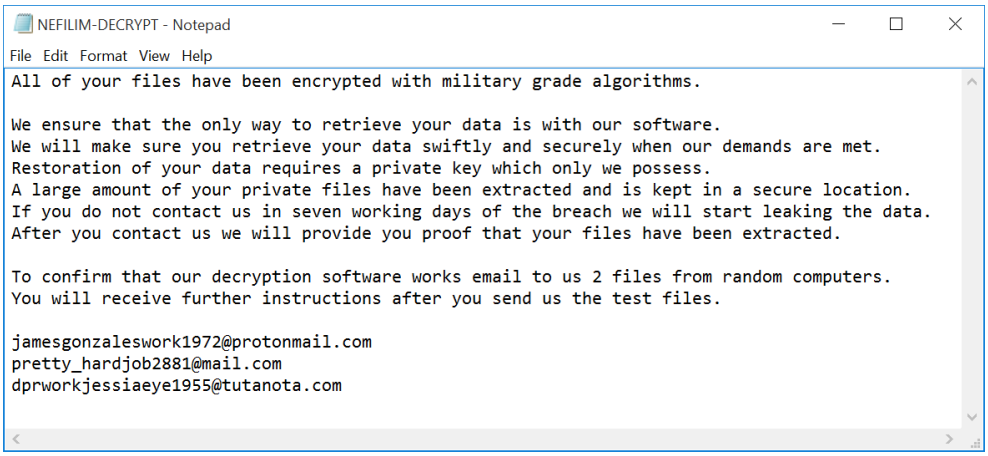
行加密，或者仅仅作为一种调试功能。

加密方式

木马的主体包含被攻击者硬编码的主 RSA-2048 公钥。在处理每个受害者的文件时，Nefilim 会生成一个 128 位的密钥和一个 128 位的 IV，并在 AES CBC 模式（密码分组链接模式）下加密文件内容。密钥和 IV 由 RSA 主密钥加密，并保存在加密文件的末尾。

被加密的文件会被重新命名，使其获得额外的扩展名.NEFILIM。

赎金记录以 NEFILIM-DECRYPT.txt 的形式保存在被处理过的目录中，并包含与勒索者联系的电子邮件地址。

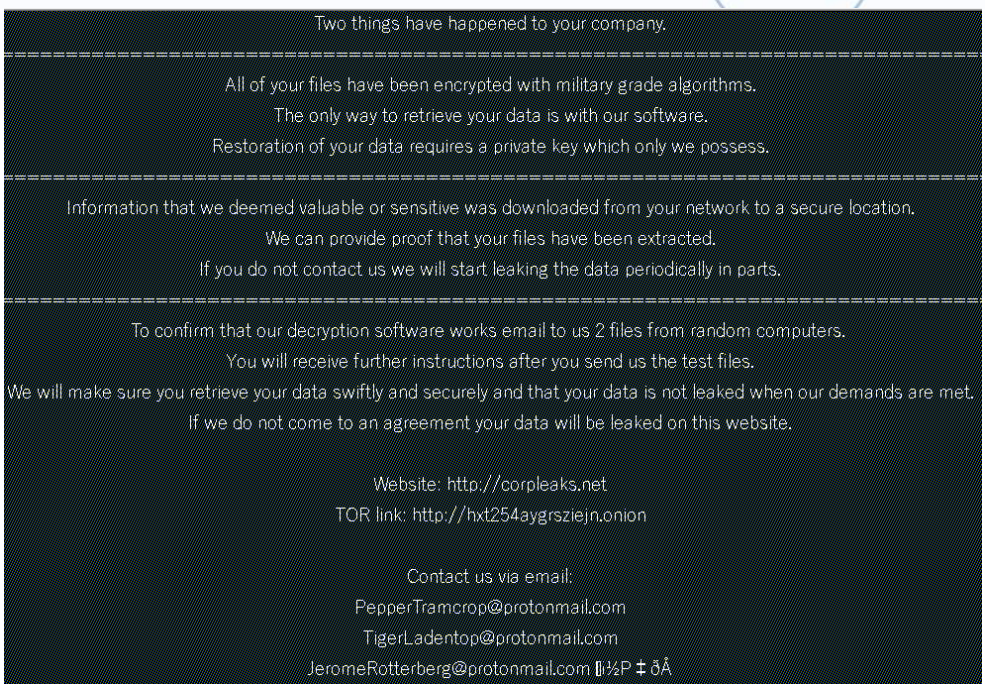


Nefilim 投放的赎金票据

2020 年 4 月：Offwhite

MD5：ad25b6af563156765025bf92c32df090

随着名称从 Nefilim 更改为 Offwhite，恶意软件的代码已被进一步修整以减小所产生的二进制大小。为了实现这一目标，开发人员停止使用 STL 库，并摆脱了不必要的 C++代码。否则，它仍然还是旧版的 Nefilim。除了我们在上文中讨论过的功能外，还有一个值得注意的功能被添加到木马代码中，即它将从勒索文本中生成壁纸并将其另存为 “scam.jpg”。



Offwhite 生成的壁纸

2020 年 6 月：Telegram

MD5：004f67c79b428da67938dadec0a1e1a4

Offwhite 和 Telegram 变体之间的差异很小，代码几乎相同，主要区别在于加密文件扩展名（.TELEGRAM）、赎金票据名称（TELEGRAM-RECOVER.txt）、以及导入的 API 函数名称没有编码为 HEX 字符串。

2020 年 11 月：Fusion

MD5：f37cebdff5de994383f34bcef4131cdf

这个木马变体是用 Go 语言编写的。正如我们上面提到的，以前的变种是用 C++开发的，这意味着可能由另一位开发者完全重写。

但是，恶意软件相似的整体操作方式、相似的加密方式、匹配的赎金记录，以及二进制文件已签名，表明这个样本实际上是 JSWorm 家族的一个新变体。

更重要的是，木马中硬编码的数据泄漏网站地址与这些攻击者之前使用的地址相同，这是一个相当有说服力的证据，支持我们的建议，即 Fusion 和它的前任者之间存在联系。

此外，像以前的变体一样，Fusion 程序接受一个命令行参数：要加密的文件名称（可能用于调试勒索软件）。

加密方式

该程序将生成两个 128 位的随机数（IV 和密钥），用于按照以下方式在 GCM 模式下使用 AES 加密文件：如果文件小于 1.5MB，则对整个文件进行加密；如果文件较大，则按顺序执行以下操作：

- 320KB 的信息被加密。
- 320KB 跳过（不加密）。
- 接下来的 320KB 被加密。
- 下一个 320KB 被跳过。
- ...
- 以此类推，直到文件末尾。

这意味着，如果是大文件，则有一半会被加密，虽然是以交替的块形式。

一个主公共 RSA 密钥被嵌入到程序中，用于加密 IV 和密钥值。一旦加密，它们就被附加到每个加密文件的末尾。

```
aBeginRsaPublic db '-----BEGIN RSA PUBLIC KEY-----',0Ah
db 'MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA4ileoxoqqU3uURadZPlK',0Ah
db 'MRZCWsnzkXNyuxYZFPDc4hREn+5kO8njounS2nRpgVTrbwMMy9bulSHOqbGGECui',0Ah
db 'gYNxSY2xiQ9tQLDDug7RAiNCw9dJnzxwkzzq+0KX+ChbQQOVMbV+FjiApEOJou8D',0Ah
db 'I9x+JlthhCGJt4oaNMV/Fnl8mLwsRLyKEC+TpBPioBoxmhNB9Rc7xuO8Mi6dg/Tf',0Ah
db 'w2A49xaCvUUvaLiCyD70IAKU12v2VerKOb1/HbkaOzgOvVdu6ekEscf9eXmO0EZ5',0Ah
db 'Sfgozun79apNaiPeVW5rvrPxhAySF4O0Yio+yjKwMYGnt7XCAE0yzNVaegoBo3sR',0Ah
db 'OQIDAQAB',0Ah
db '-----END RSA PUBLIC KEY-----',0Ah,0
```

Fusion 使用的 RSA 公钥

最后，“FUSION ”行被写在文件的末尾。之后，扩展名“.FUSION ”被附加到文件名上。该样本还留下了一个带有通信联系方式的说明（FUSION-README.txt）：

```
Two things have happened to your company.
==========================================================================================================================
Gigabytes of archived files that we deemed valuable or sensitive were downloaded from your network to a secure location.
When you contact us we will tell you how much data was downloaded and can provide extensive proof of the data extraction.
You can analyze the type of the data we download on our websites.
If you do not contact us we will start leaking the data periodically in parts.
==========================================================================================================================
We have also encrypted files on your computers with military grade algorithms.
If you don't have extensive backups the only way to retrieve your data is with our software.
Restoration of your data with our software requires a private key which only we possess.
==========================================================================================================================
To confirm that our decryption software works send 2 encrypted files from random computers to us via email.
You will receive further instructions after you send us the test files.
We will make sure you retrieve your data swiftly and securely and your data that we downloaded will be securely deleted when our demands are met.
If we do not come to an agreement your data will be leaked on this website.

Website: http://[redacted].net
TOR link: http://[redacted].onion

Contact us via email:
idahaines2020@tutanota.com
kristenjones25@tutanota.com
joycepmills@protonmail.com
```

Fusion 投放的赎金票据

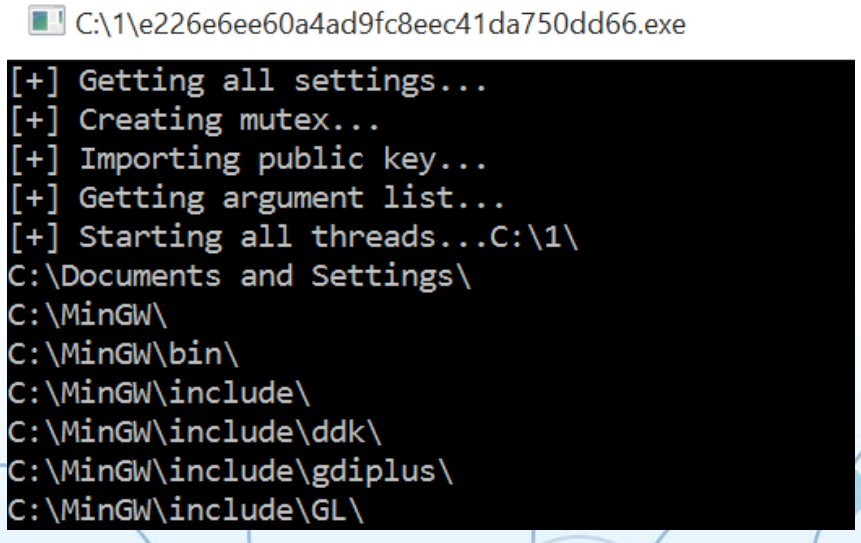
2021 年 1 月：Milihpen

MD5：e226e6ee60a4ad9fc8eec41da750dd66

随着 Milihpen 变体的出现，JSWorm 家族背后的攻击者再次完全重写了恶意软件的代码，或者说是雇佣了另一个开发者从头开始编写。这个样本再次使用 C++开发（像 Nefilim 和以前的变种），而不是使用 Golang（像 Fusion）。

尽管如此，仍保留了主要功能、执行流程、加密方式和数据泄漏网站地址。此外，木马的名字显示了与以前的一个恶意软件变体的联系，即它是“Nephilim”一词的倒写。

该木马现在将其所有操作记录到控制台，这可能使恶意软件背后的攻击者更方便地控制感染过程。



Milihpen 的控制台日志记录

与以前的变体一样，恶意软件样本由数字证书签名。启动后，Milihpen 解析了木马主体中硬编码的配置数据。该配置结构以 JSON 格式存储，包含以下字段:

```
    {
        //mutex name
        "mutex": "MILIHPEN",

        //encrypted file extension
        "ext": "MILIHPEN",

        //ransom note name part
        "nt_name": "-INSTRUCT.txt",

        //master RSA public key (base64-encoded), redacted
        "pub": "UlNB...Bnum9ew==",

        //ransom note text (base64-encoded), redacted
        "nt_content": "VHdvIHRoa...wuY29t",

        //skipped file extensions
        "whiteext": [".exe", ".dll", ".lnk", ".url", ".log", ".cab", ".cmd",
".bat", ".dll", ".msi", ".mp3", ".mp4", ".pif", ".ini"],

        //skipped directory names
        "whitedir": ["windows", "programdata", "program files", "program files
(x86)", "appdata", "$recycle.bin", "all users", ".", "..", "rsa"],

        //dynamically imported API function names
        "winapi": ["MessageBoxA", "MessageBoxW", "BCryptOpenAlgorithmProvider",
"BCryptGenRandom", "BCryptImportKeyPair", "BCryptEncrypt"]
    }
```

在解析了配置中的值之后，Milihpen 创建了一个 mutex，解析了命令行参数，并继续以与 Nefilim 和最近的 JSWorm 变体相同的逻辑进行操作。如果提供了命令行参数，该木马会检查它是否是一个目录路径。如果是，它将对其中的文件进行加密；否则，它将把它解释为一个文件路径，并尝试对其进行加密。如果没有提供参数，木马会在所有本地和远程驱动器中搜索受害者的文件。

```
do
{
  v6 = (v5++)["\"mutex\":\"MILIHPEN\", \"ext\":\"MILIHPEN\", \"nt_name\":\"-INSTRUCT.txt\", \"pub\":\"U1NBMQAIAAADAAAAA"
              "AEAAAAAAAAAAAAAAQABifpIEBtVtSKRuuwYI+oiFv7nX3XurT35vPt7X1lgPuZFZVhfiAJTEXCbs0y2f68jB60pv4AYNsi0THhQgdn2l"
              "Gf3T3V11f2j87cOdkwpQQpduf9BTggVHeEx04WnFnL5UJOELgK13c1VxNS1OpeyYJo+kncDjQrlGR/R2VRtE4/bzkhll9HVEEJS8J04+"
              "72C53ZHWgApPX3vPrOfyU9Uj0CRSojD3lnCLxd77coDfjttmshmKy1f2cEz5k4VeCqiWbjMVP0shRFTTOvdZJY9J4Fuw85HfkbkPG2Gf"
              "GMfzqHj4U1PZVCnBJ3ZWE4SQRNDfUftIyHaaMtUKT0Bnum9ew==\", \"nt_content\":\"VHdvIHRoaW5ncyBoYXZlIGhhcHBlbmVk"
              "IHRvIHlvdXIgY29tcGFueS4NCj09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09"
              "PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09"
              "YXJjaG12ZWQgZmlsZXMgdGhhdCB3ZSBkZWVtZWQgdmFsdWFibGUgb3Igc2Vuc210aXZlIHdlcmUgZG93bmxvYWRlZCBmcm9tIHlvdXIg"
              "bmV0d29yay80byBhIHNlY3VyZSBsb2NhdGlvbi4NCldoZW4geW91IGNvbnRhY3QgdXMgd2Ugd2lsbCB0ZWxsIHlvdSBob3cgbXVjaCBk"
              "YXRhIHdhcy Bkb3dubG9hZGVkIGFuZCBjYW4gcHJvdmlkZSBleHRlbnNpdmUgcHJvb2Ygb2YgdGhlIGRhdGEgZXh0cmFjdGlvbi4gDQpZ"
              "b3UgY2FuIGFuYWx5emUgdGhlIHR5cGUgb2YgdGhlIGRhdGEgd2UgG93J3QgaGF2ZSBhbHNvIGVuY3J5cHRlZCBmaWxlcyBvbiB5b3VyIGNvbXB1"
              "cyBzZW5kIDIgZW5jcnlwdGVkIGZpbGVzIGZyb20gcmFuZG9tIGNvbXB1dGVycy B0byB1cyB2aWEgZW1haWwuDQpZb3Ugd2lsbCByZWNl"
              "aXZlIGZ1cnRoZXIgaW5zdHJ1Y3Rpb25zIGFmdGVyIHlvdSBzZW5kIHVzIHRoZS B0ZXN0IGZpbGVzLg0KV2Ugd2lsbCBtYWtlIHN1cmUg"
              "eW91IHJldHJpZXZlIHlvdXIgZGF0YSBzd2lmdGx5IGFuZCBzZWN1cmVseSBhbmQgeW91ciBkYXRhIHRoYXQgd2UgZG93bmxvYWRlZCB3"
              "aWxsIGJlIHNlY3VyZWx5IGRlbGV0ZWQgd2hlbiBvdXIgZGVtYW5kcyBhcmUgbWV0Lg0KSWYgd2UgZG8gbm90IGNvbWUgdG8gYW4gYWdy"
              "ZWVtZW50IHlvdXIgZGF0YSB3aWxsIGJlIGxlYWtlZCBvbi B0aGlzIHdlYnNpdGUuDQoNCldlYnNpdGU6IGh0dHA6Ly9jb3JwbGVha3Mu"
              "bmV0DQpUT1IgbGluazogaHR0cDovL2h4dDI1NGF5Z3Jzemllam4ub25pb24NCg0KQ29udGFjdCB1cyB2aWEgZW1haWw6DQptYXJrdXNw"
              "ZWllycmVyZWExNzdAdHV0YW5vdGEuY29tDQpnaW9tYXJrdXNuaWVsc29uQHR1dGFub3RhLmNvbQ0KbWFya3VzcGVpcnJlcmVhMTc3QHBy"
              "b3Rvbm1haWwuY29t\", \"whiteext\":[\".exe\",\".dll\",\".lnk\",\".url\",\".log\",\".cab\",\".cmd\",\".bat\""
              ",\".dll\",\".msi\",\".mp3\",\".mp4\",\".pif\",\".ini\"], \"whitedir\":[\"windows\",\"programdata\",\"pro"
              "gram files\",\"program files (x86)\",\"appdata\",\"$recycle.bin\",\"all users\",\".\",\"..\",\"rsa\"], \""
              "winapi\":[\"MessageBoxA\",\"MessageBoxW\",\"BCryptOpenAlgorithmProvider\",\"BCryptGenRandom\",\"BCryptIm"
              "portKeyPair\",\"BCryptEncrypt\"]}"
          - v4];
  *v5 = v6;
}
while ( v6 );
*(_OWORD *)v12 = 0i64;
v13 = 0i64;
g_json = ParseJson(v4, (int)v12);
if ( !g_json )
{
  Log("[-] Failed to parse JSON... exit\n");
  Sleep(5000u);
  ExitProcess(1u);
}
```

Milihpen 样本中的配置 JSON 结构

加密方式

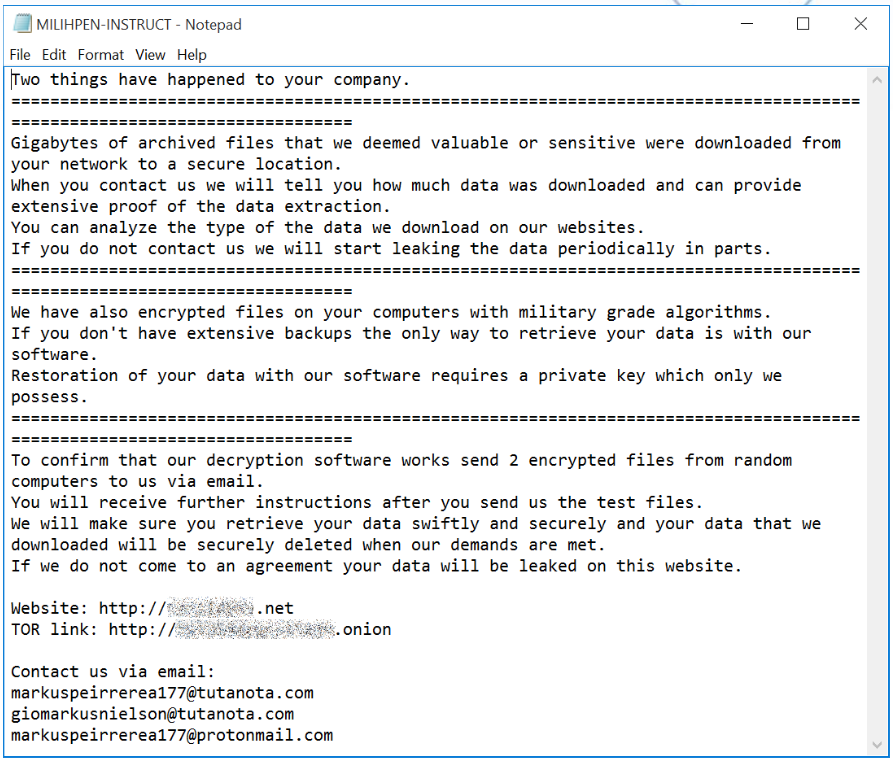
与其它方面一样，Milihpen 模仿了从 Nefilim 开始的以前的变体及后续版本变体的高级逻辑。但是，代码分析表明其功能的实现已被完全重写。

为了加密文件，Milihpen 使用相同的算法：CBC 模式的 AES 和 RSA。AES 密钥和 IV 也是 128 位的，并在加密后由主公共 RSA 密钥保存在加密文件的末尾。

对于随机数生成和 RSA 加密，与以前的版本不同，Milihpen 使用 BCrypt API 的函数，它是 Windows Vista 中引入的 Cryptography Next Generation（CNG）API 的一部分，这并没有为 Milihpen 提供任何明显的优势，但这是一个值得注意的特征，因为 Bcrypt API 现在并不常用于加密勒索软件中。

加密的文件被重新命名为一个附加的扩展名.MILIHPEN，赎金记录被保存为 MILIHPEN-INSTRUCT.txt。

赎金记录包含与这个家族以前的变体类似的文字，以及相同的数据泄漏网站地址。



MILIHPEN-INSTRUCT - Notepad

File Edit Format View Help

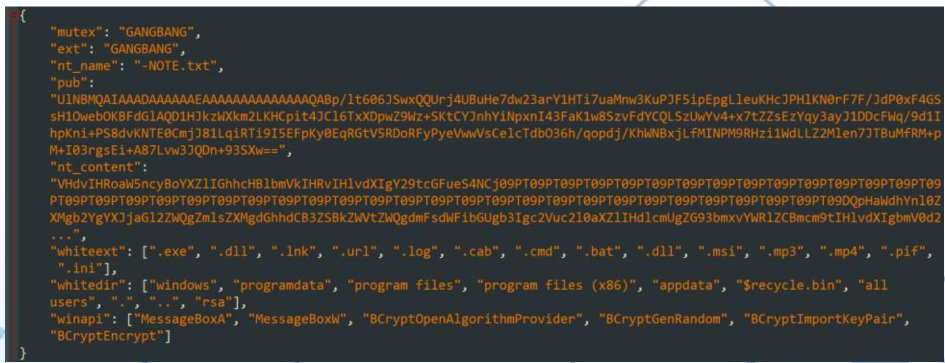
```
Two things have happened to your company.
===========================================================================================================================
Gigabytes of archived files that we deemed valuable or sensitive were downloaded from
your network to a secure location.
When you contact us we will tell you how much data was downloaded and can provide
extensive proof of the data extraction.
You can analyze the type of the data we download on our websites.
If you do not contact us we will start leaking the data periodically in parts.
===========================================================================================================================
We have also encrypted files on your computers with military grade algorithms.
If you don't have extensive backups the only way to retrieve your data is with our
software.
Restoration of your data with our software requires a private key which only we
possess.
===========================================================================================================================
To confirm that our decryption software works send 2 encrypted files from random
computers to us via email.
You will receive further instructions after you send us the test files.
We will make sure you retrieve your data swiftly and securely and your data that we
downloaded will be securely deleted when our demands are met.
If we do not come to an agreement your data will be leaked on this website.

Website: http://[illegible].net
TOR link: http://[illegible].onion

Contact us via email:
markuspeirrerea177@tutanota.com
giomarkusnielson@tutanota.com
markuspeirrerea177@protonmail.com
```

Milihpen 投放的赎金票据

2021 年 2 月：Gangbang

MD5：173ab5a59490ea2f66fe37c5e20e05b8

Gangbang 变体与 Milihpen 相同，是该勒索软件家族中最近发现的变体。唯一值得注意的区别是，配置结构现在由 AES 用硬编码的密钥和 IV 进行加密，而不是像 Milihpen 那样采用明文形式。此外，与以前的版本相比，这个样本上的数字签名是无效的。



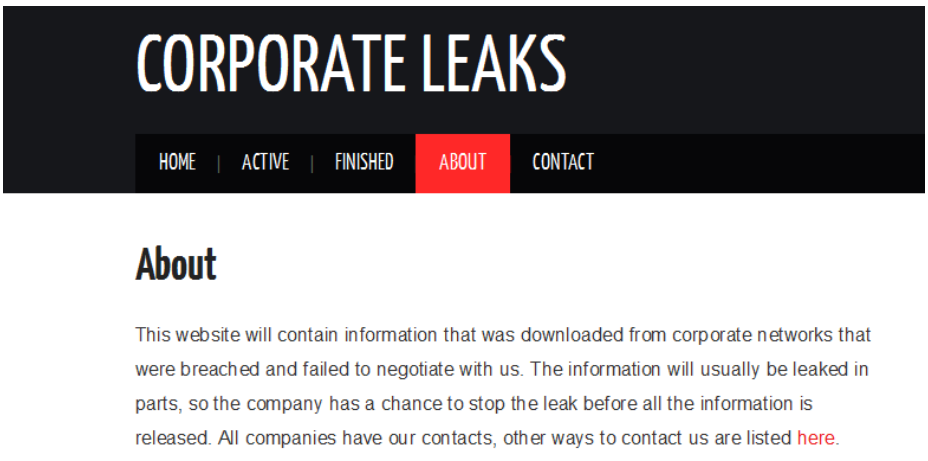

```
{
    "mutex": "GANGBANG",
    "ext": "GANGBANG",
    "nt_name": "-NOTE.txt",
    "pub":
    "UlNBMQAIAAADAAAAAAEAAAAAAAAAAAAAAQABp/lt606JSwxQQUrj4UBuHe7dw23arY1HTi7uaMnw3KuPJF5ipEpgLleuKHcJPHlKN0rF7F/JdP0xF4GS
    sH1OwebOKBFdGlAQD1HJkzWXkm2LKHCpit4JCl6TxXDpwZ9Wz+SKtCYJnhYiNpxnI43FaK1w8SzvFdYCQLSzUwYv4+x7tZZsEzYqy3ayJ1DDcFWq/9d1I
    hpKni+PS8dvKNTE0CmjJ81LqiRTi9I5EFpKy0EqRGtV5RDoRFyPyeVwwVsCelcTdbO36h/qopdj/KhWNBxjLfMINPM9RHzi1WdLLZ2Mlen7JTBuMfRM+p
    M+I03rgsEi+A87Lvw3JQDn+93SXw==",
    "nt_content":
    "VHdvIHRoaW5ncyBoYXZlIGhhcHBlbmVkIHRvIHlvdXIgY29tcGFueS4NCj09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09
    PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09PT09DQpHaWdhYnl0Z
    XMgb2YgYXJjaGl2ZWQgZmlsZXMgdGhhdCB3ZSBkZWVtZWQgdmFsdWFibGUgb3Igc2Vuc2l0aXZlIHdlcmUgZG93bmxvYWRlZCBmcm9tIHlvdXIgbmV0d2
    ...",
    "whiteext": [".exe", ".dll", ".lnk", ".url", ".log", ".cab", ".cmd", ".bat", ".dll", ".msi", ".mp3", ".mp4", ".pif",
     ".ini"],
    "whitedir": ["windows", "programdata", "program files", "program files (x86)", "appdata", "$recycle.bin", "all
    users", ".", "..", "rsa"],
    "winapi": ["MessageBoxA", "MessageBoxW", "BCryptOpenAlgorithmProvider", "BCryptGenRandom", "BCryptImportKeyPair",
    "BCryptEncrypt"]
}
```


Gangbang 样本的配置（片段）

数据泄露网站

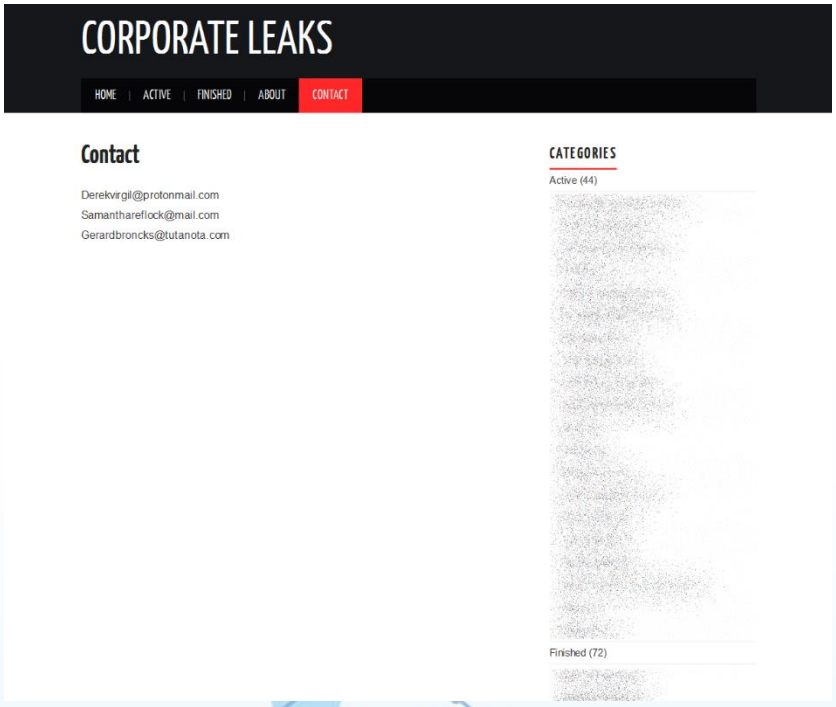
2020 年春，JSWorm 家族背后的攻击者转而从事大型狩猎活动，并建立了自己的网站，他们可以在其中发布从受害者那里窃取的机密数据。

在撰写本文时，该网站仍在运行，并包含一百多个成为该恶意软件受害者的组织的帖子。



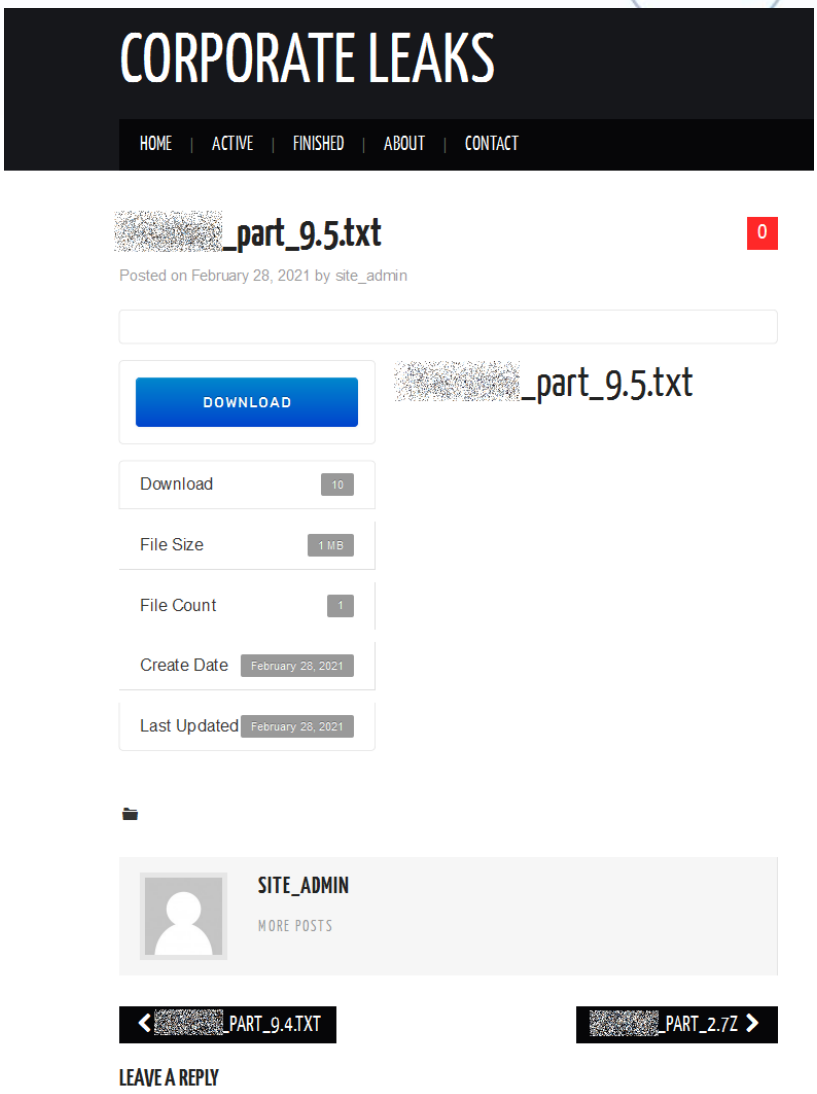
攻击者的说明页面

“contact”页面列出了攻击者当前用于谈判的电子邮件地址：



包含联系人电子邮件地址的页面

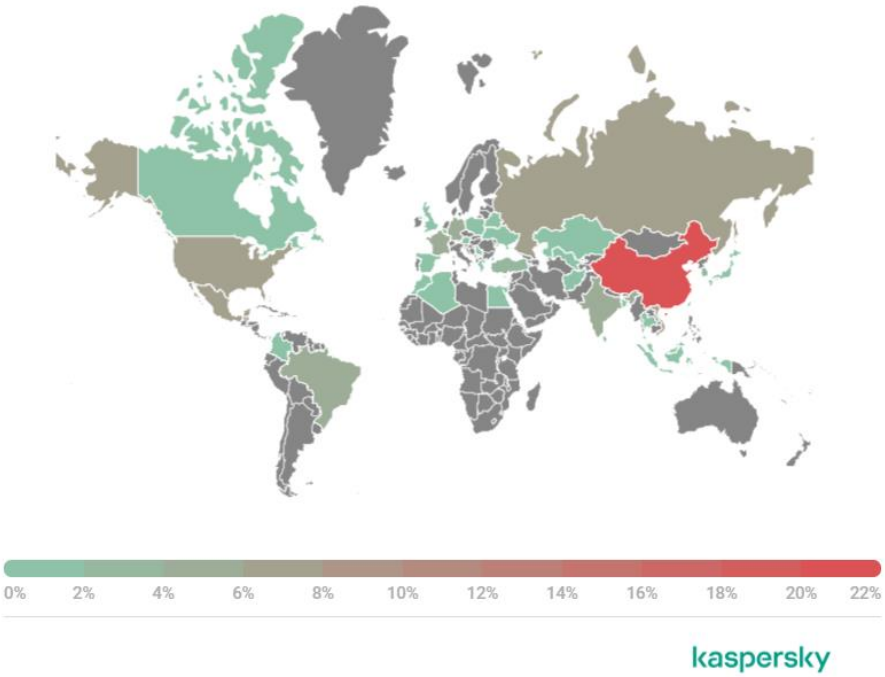
有些受害者也有单独的页面，可以下载他们的一些被盗数据。



被盗数据下载页面

受害者

我们根据 KSN 遥测数据创建了一个图表，说明了 JSWorm 勒索软件攻击的地理分布。



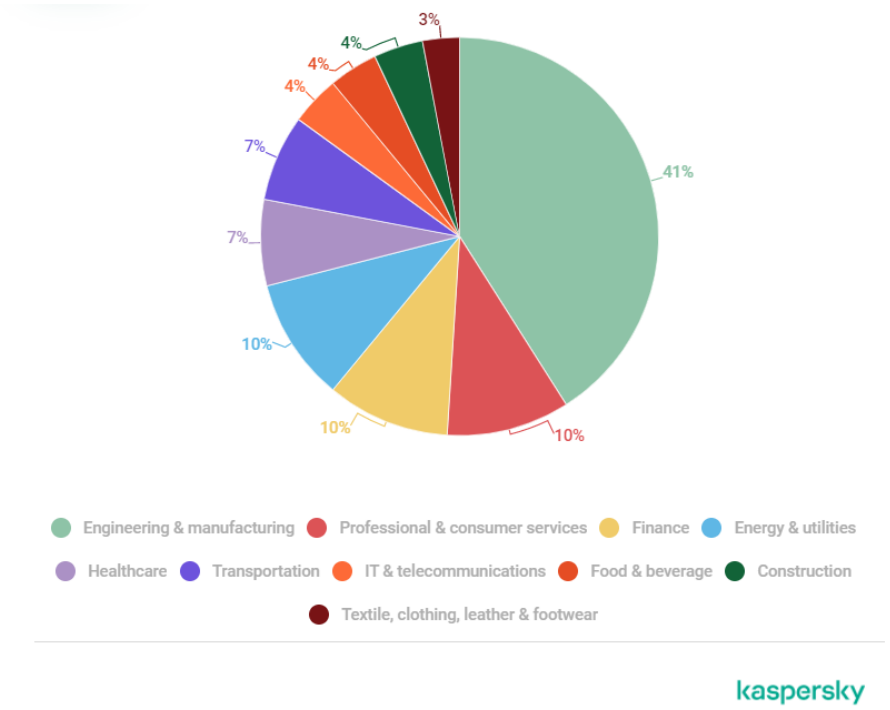
JSWorm 受害者的地理分布情况（根据 KSN 遥测数据）

根据 KSN 统计，被 JSWorm 攻击的前 10 个国家：

序列	国家	%*
1	中国	21.39%
2	美国	7.96%
3	越南	7.46%
4	墨西哥	6.97%
5	俄罗斯联邦	6.47%
6	巴西	5.47%
7	印度	5.47%
8	德国	4.98%
9	法国	4.48%
10	大韩民国	2.99%

受 JSWorm 勒索软件家族攻击的用户占受害者总数的百分比（基于国家）

我们还分析了攻击者在其数据泄漏网站上发布的有关受害者的数据，根据这些数据，我们创建了一个图表，说明了 JSWorm 受害者的行业分布情况。



JSWorm 受害者的行业分布（根据攻击者的数据泄漏网站上的数据）

根据攻击者公布的受害者名单，五分之二（41%）的 JSWorm 攻击是针对工程和制造类公司的。能源和公用事业（10%）、金融（10%）、专业和消费者服务（10%）、运输（7%）和医疗保健（7%）也在他们的名单上名列前茅。

结论

JSWorm 家族已经发展了两年，在此期间，它改变了分发模式，木马经历了几次完整的重新开发。自 2019 年发现之后，它已经从一个典型的大规模勒索软件威胁（主要影响个人用户）变成了一个典型的大型目标勒索软件威胁（攻击知名度高的目标，并要求支付大量赎金）。

与当今其它有针对性的勒索软件威胁一样，防止 JSWorm 感染事件的关键是采取较为完善的防护措施来保护组织的网络安全。任何弱点都可能成为攻击者的入口，例如存在漏洞的

旧版服务器端软件、员工点击恶意链接、远程控制系统的弱口令等等。

为了加强对大型目标勒索软件的防御，我们建议定期对网络进行安全审计，以及时发现并修复安全漏洞。

最大限度地提高组织安全性的其它建议：

- 非必要时，不要将远程桌面服务（如 RDP）暴露在公共网络中，并始终对其使用强密码。
- 确保商业 VPN 解决方案和其它服务器端软件始终是最新的，因为它们是勒索软件的常见感染载体。另外，始终保持客户端的应用程序也是最新的。
- 将防御策略集中在检测横向移动和数据渗出到互联网上，要特别注意传出的流量，以检测网络犯罪的连接。定期备份数据，确保在需要时可以在紧急情况下快速访问它。使用最新的威胁情报信息，以实时了解攻击者所使用的 TTP。
- 使用卡巴斯基端点检测和响应以及卡巴斯基管理检测和响应服务等解决方案，帮助在攻击者实现其最终目标之前，在早期阶段识别并阻止攻击。
- 为了保护企业环境，对员工进行安全教育。专门的培训课程可以提供帮助，如卡巴斯基自动安全意识平台中提供的课程。
- 使用可靠的端点安全解决方案，如卡巴斯基企业端点安全，它由漏洞预防、行为检测和可回滚恶意行为的修复引擎驱动。KESB 还具有自我防御机制，可以防止其被网络犯罪分子删除。

IOC

JSWorm（早期版本）

MD5: a20156344fc4832ecc1b914f7de1a922

JSWorm 4.0.3

MD5: 5444336139b1b9df54e390b73349a168

Nemty 1.4

MD5: 1780f3a86beceb242aa81afecf6d1c01

Nefilim

MD5: 5ff20e2b723edb2d0fb27df4fc2c4468

Offwhite

MD5: ad25b6af563156765025bf92c32df090

Telegram

MD5: 004f67c79b428da67938dadec0a1e1a4

Fusion

MD5: f37cebdff5de994383f34bcef4131cdf

Milihpen

MD5: e226e6ee60a4ad9fc8eec41da750dd66

Gangbang

MD5: 173ab5a59490ea2f66fe37c5e20e05b8

原文连接：

https://securelist.com/evolution-of-jsworm-ransomware/102428/