【工具】微软 sysmon 使用总结 - Welcome to Security WorldWelcome to Security World

> 0x00 前言 sysmon 相信经常处理 windows 应急响应的朋友都不陌生了，这款强大的轻量级监控工具。之前 [...]

sysmon 相信经常处理 windows 应急响应的朋友都不陌生了，这款强大的轻量级监控工具。之前在没有思路的时候使用它监控 windows 各种行为，会有意想不到的收获。

借用 freebuf 上的介绍：

sysmon 是由 Windows Sysinternals 出品的一款 Sysinternals 系列中的工具。它以系统服务和设备驱动程序的方法安装在系统上，并保持常驻性。sysmon 用来监视和记录系统活动，并记录到 windows 事件日志，可以提供有关进程创建，网络链接和文件创建时间更改的详细信息。

通过收集使用 Windows 事件集合或 SIEM 代理生成的事件，然后分析它们，可以识别恶意或异常活动，并了解入侵者和恶意软件在您的网络上如何操作。

为什么这次要写这篇文章，主要是因为最近，这工具居然更新了 DNS 查询记录功能，让我们可以监控到系统上进程的 DNS 查询日志。这可是一个超级实用的功能，在很多时候，在一些检测设备，如 IDS, 态势感知上发现有恶意域名解析，这种在终端查起来是比较麻烦的，因为你只有这么一个线索，而且 DNS 查询这种流量，你要从他定位到恶意进程，还是有难度的；也有其他方法定位，但没有 sysmon 这么吊，比如可以通过内存扫描的方式，扫描域名字符串，过滤出匹配到的进程，然后

再进一步用如 processhacker 工具去定位内存字符串是否确实存在，但这种方式只能在病毒还运行的时候能查到，可能潜伏没运行，并且这个字符串得还留存在内存中；或者用科来系统也可以监控数据包并锁定进程，但科来系统太大了，并且抓包也无法长期运行，sysmon 轻便并且可以长期监控系统。除此之外我还想了一些歪门邪道，比如我写的一个工具 domaintoprocess，但说实话不怎么好用，或者就自己开发，hook 像 gethostbyname 一类解析域名的 API，其实这个效果就和现在的 sysmon 差不多，它既然有了就省事很多了。

所以在此写一写自己的一些使用心得吧。

要使用该工具，首先要进行安装，该工具安装后会以服务及驱动的形式运行，监控系统，经过测试这工具只支持 win2008 以上版本，这是一个缺憾。

安装时要使用管理员权限运行 cmd，最常用的安装方式就是

sysmon.exe –accepteula –i –n

–i 即 install，–n 会监听 network 连接，sysmon 有 32 位和 64 位两种，根据系统选择。

但我在此推荐大家使用另一个方式安装，sysmon 可以使用 xml 配置文件进行安装，需要先配置好 xml 文件，但大家不必担心 xml，这个已经有大佬为大家提供一个推荐的 xml 配置文件，并可以在此基础上进行修改。该配置文件为大家过滤了一些不必要监控的系统行为以及选择捕捉适当的条目，可以在应急响应中，将注意力集中真正有意义的日志上，并尽可能减少性能影响。

配置文件下载链接如下

https://github.com/SwiftOnSecurity/sysmon-config

安装命令如下

Sysmon64.exe –accepteula –i z-AlphaVersion.xml

后面带的是最近拿到的可以监听 dns 查询的 xml 配置文件，不过还是 alpha 版，需要完善。

安装后，如果需要修改配置文件，可使用以下命令修改

sysmon.exe –c sysmonconfig–export.xml

如果需要卸载使用以下命令

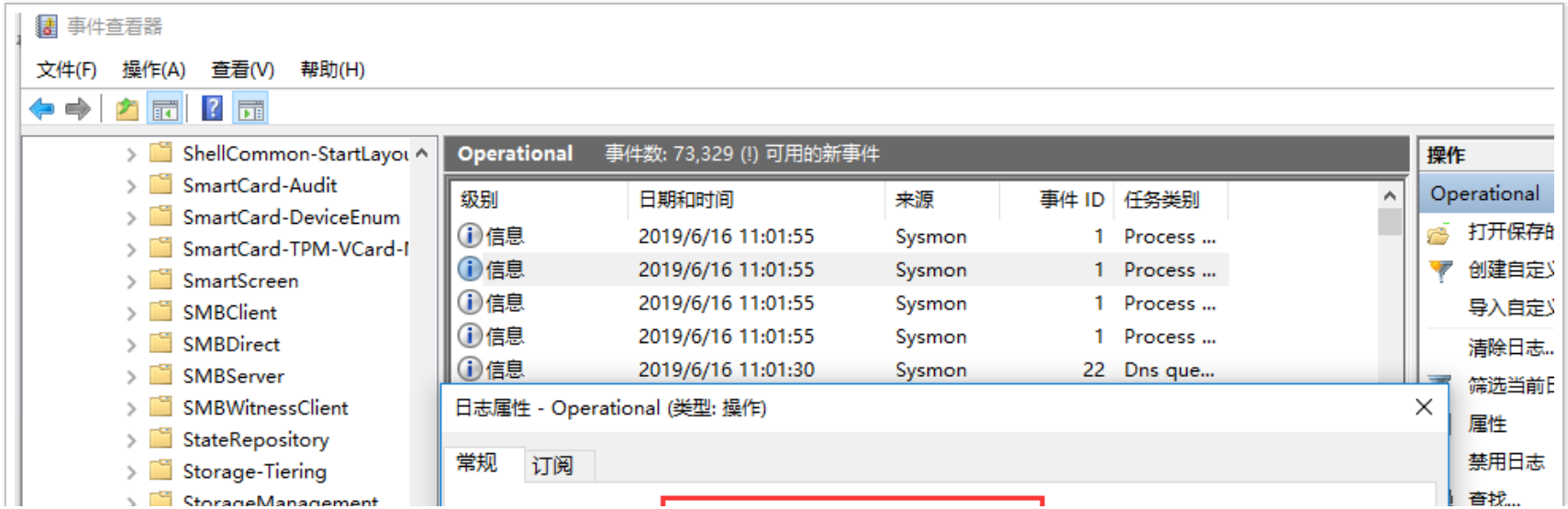
sysmon.exe –u

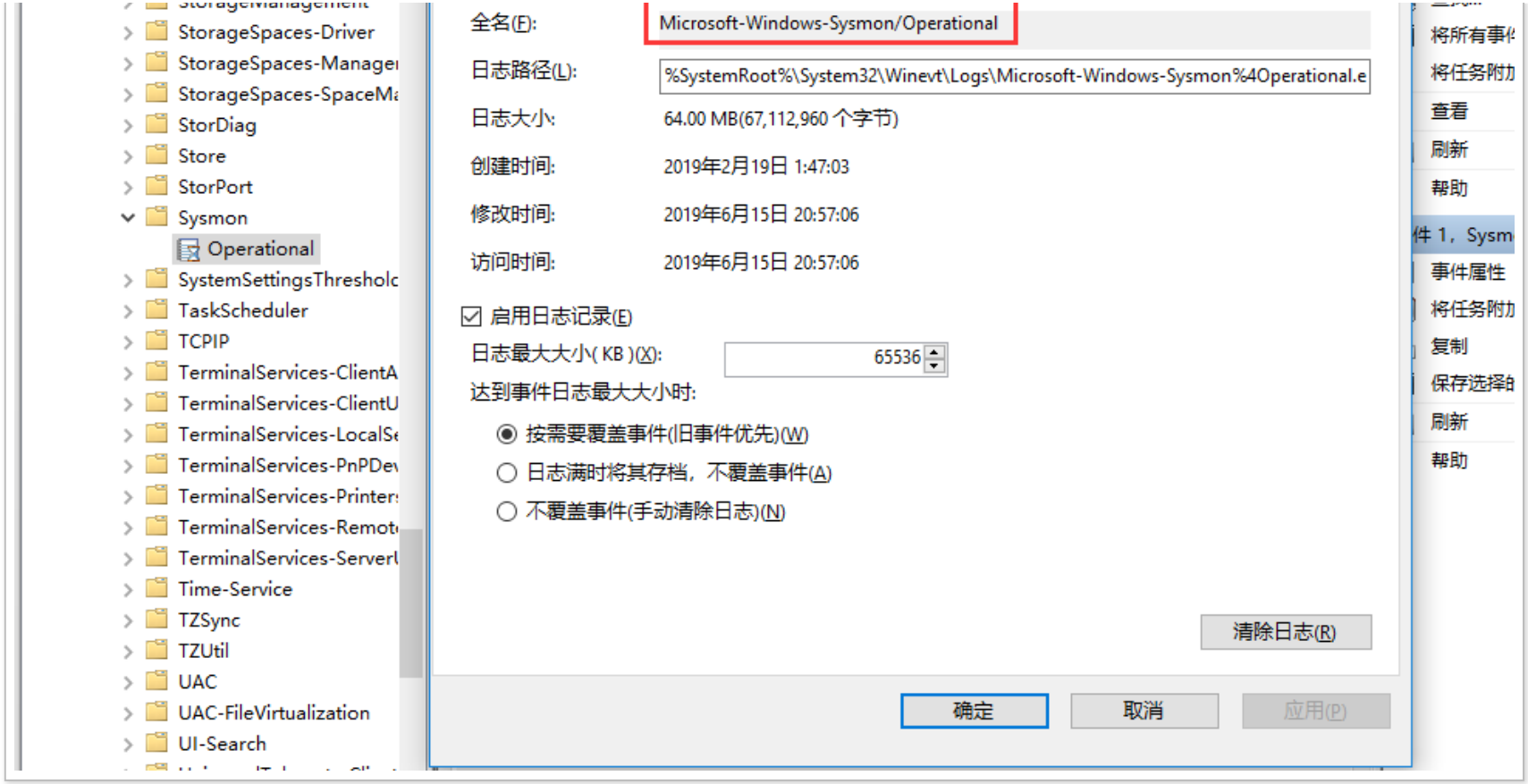
日志通过事件查看器查看，因为 sysmon 的日志是以 evtx 格式存储的。

具体事件路径为

应用程序和服务日志 – Microsoft–Windows–Sysmon–Operational

如下图所示，或者你直接去 C 盘指定路径查文件也行





如同 windows 自带的系统日志，安全日志有事件 ID 一样，sysmon 日志也有对应的事件 ID，最新版本支持 23 种事件。

Event ID 1: Process creation

Event ID 2: A process changed a file creation time

Event ID 3: Network connection

Event ID 4: Sysmon service state changed

Event ID 5: Process terminated

Event ID 6: Driver loaded

Event ID 7: Image loaded

Event ID 8: CreateRemoteThread

Event ID 9: RawAccessRead

Event ID 10: ProcessAccess

Event ID 11: FileCreate

Event ID 12: RegistryEvent (Object create and delete)

Event ID 13: RegistryEvent (Value Set)

Event ID 14: RegistryEvent (Key and Value Rename)

Event ID 15: FileCreateStreamHash

Event ID 17: PipeEvent (Pipe Created)

Event ID 18: PipeEvent (Pipe Connected)

Event ID 19: WmiEvent (WmiEventFilter activity detected)

Event ID 20: WmiEvent (WmiEventConsumer activity detected)

Event ID 21: WmiEvent (WmiEventConsumerToFilter activity detected)

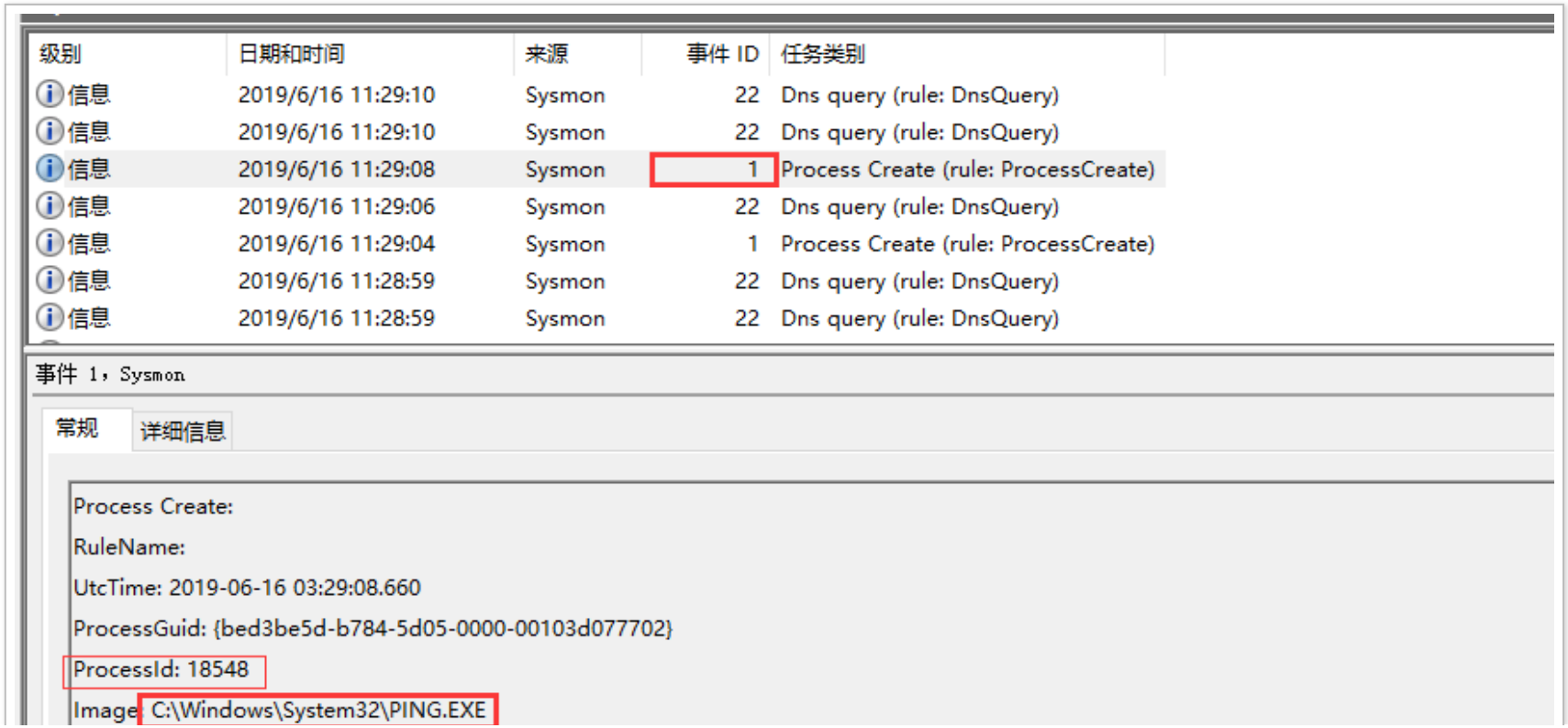
Event ID 22: DNSEvent (DNS query)

Event ID 255: Error

常用的有事件 ID 1，监控进程创建，恶意进程的创建，包括他的父进程，PID，执行命令等等。

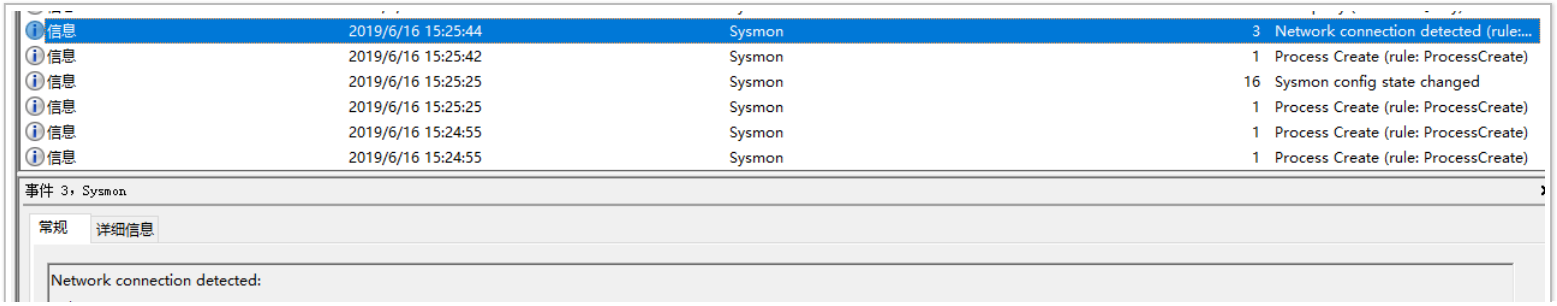
之前遇到过一起，开启会自动运行 powershell，但查了启动项，任务计划，wmi 都没发现痕迹，苦苦无解，然后使用 sysmon 监控进程创建，最终定位是谁拉起了 powershell，往上溯源找到是一个伪造成正常程序图标和后缀的 link 文件，存放在 Startup 目录下，链接到存放在另一处的 vbs 脚本。

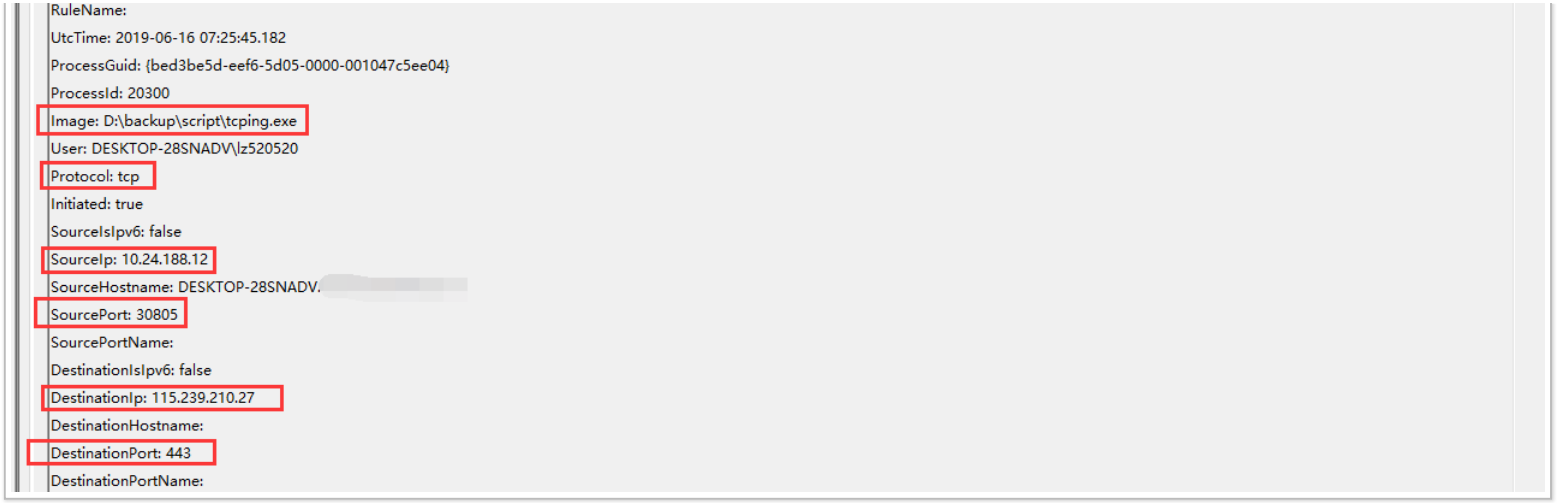
下图即为进程创建监控日志，可以看到几个关键点，创建的进程，命令行，以及父进程。


级别 日期和时间 来源 事件 ID 任务类别
信息 2019/6/16 11:29:10 Sysmon 22 Dns query (rule: DnsQuery)
信息 2019/6/16 11:29:10 Sysmon 22 Dns query (rule: DnsQuery)
信息 2019/6/16 11:29:08 Sysmon 1 Process Create (rule: ProcessCreate)
信息 2019/6/16 11:29:06 Sysmon 22 Dns query (rule: DnsQuery)
信息 2019/6/16 11:29:04 Sysmon 1 Process Create (rule: ProcessCreate)
信息 2019/6/16 11:28:59 Sysmon 22 Dns query (rule: DnsQuery)
信息 2019/6/16 11:28:59 Sysmon 22 Dns query (rule: DnsQuery)
事件 1，Sysmon
常规
详细信息
Process Create:
RuleName:
UtcTime: 2019-06-16 03:29:08.660
ProcessGuid: {bed3be5d-b784-5d05-0000-00103d077702}
ProcessId: 18548
Image: C:\Windows\System32\PING.EXE

事件 ID3，监控网络连接，当恶意程序外连 CC 服务器或者矿池等操作的时候，可以监控到是哪个进程发起的连接。这边也举个例子，之前遇到一种病毒，当你去用进程管理器或分析工具去查看时，该病毒会自动退出，防止被检测到，并且随机一段时间重启，但态势感知上发现确实有挖矿行为，使用 sysmon 监控，当他不定时运行时，即可捕捉到他连接矿池的行为，从而定位到进程。

下图为网络连接监控日志，可以看到网络连接的五元组，和对应的进程。

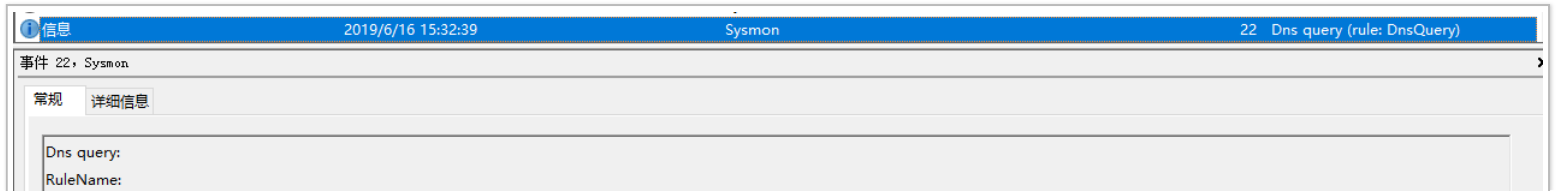




事件 ID22，是这次重磅推出的新功能，DNS 查询记录，这功能让应急响应人员可以很轻松的通过域名定位到进程，并且你即使开启了 dnscache 服务，也能定位到原先进程，sysmon 原理没有分析过，猜测是监控了一些跟域名解析相关的 API，而 dnscache 是这些 API 调用后下一步才会用到的，所以先于 dnscache 监控到原始进程。dnscache 是一个缓存服务，简单来讲，就是会代理其他进程进行 dns 解析，并且会缓存解析结果，下一次解析就不再发送请求，读取缓存内容返回给指定程序即可。所以大家使用内存扫描工具，可能会定位到 dnscache 服务进程。

在监测设备上发现可疑域名解析时，通过这个功能，可以轻松定位到发起解析的进程，从而进一步分析进程文件是否确实有问题。

如下图所示，为 dns 查询日志，会记录解析域名和结果，以及对应的进程 PID 和路径。



我就举几个常用的事件类型来讲解，其他的事件类型我后续再对应案例具体分析吧，大家可以参考文章最后面 freebuf 链接以及微软链接，会有比较详细的说明。

其实 sysmon 最大优势就是可以实时监控，往往会遇到这么一个情况，检测设备检测到可疑行为，会有一个时延，而当工程师去查看设备时，又有一个时延，所以经常发现病毒的时候，已经滞后很长时间了，这个时候再去排查，可能病毒不在运行了，杀软又扫描不出来，无从下手，实际上也确实没有比较好的方法，虽然通过各种手段尝试或许能找到，比如查看可疑服务，任务计划，逐条分析排查，但需要消耗的时间会比较大，这个时候其实可以以使用 sysmon 进行后续监控，当再一次发现的时候进行日志分析处理。一来在没有思路的时候，可以以这个方式来缓冲一下，给自己留点时间思考分析，或者和其他人讨论下有没遇到过，或者思路啥的。二来，确实觉得这个时候没法查下去，或再查下去很费时间，用户可能办公也还要使用主机，但也不能跟用户说查不到没办法，这个时候就可以以此来为这次排查做个收尾闭环，并给出下一步计划，说明这次排查分析没发现病毒运行痕迹，可能处于潜伏，但我们在系统上安装了监控，后续再次发现问题的时候，我们可以提取日志分析定位问题，这样让用户知道当前进度以及困难之处，和下一步的安排。

0x02 配置文件

上面说了如何使用以及各类日志和应用场景，并且提到使用推荐的 xml 文件进行过滤，因为这个 xml 文件其实是一个推荐模板，可能会需要根据具体场景进行修改，下面就分析下这个 xml 文件，以及如何修改满足需求。

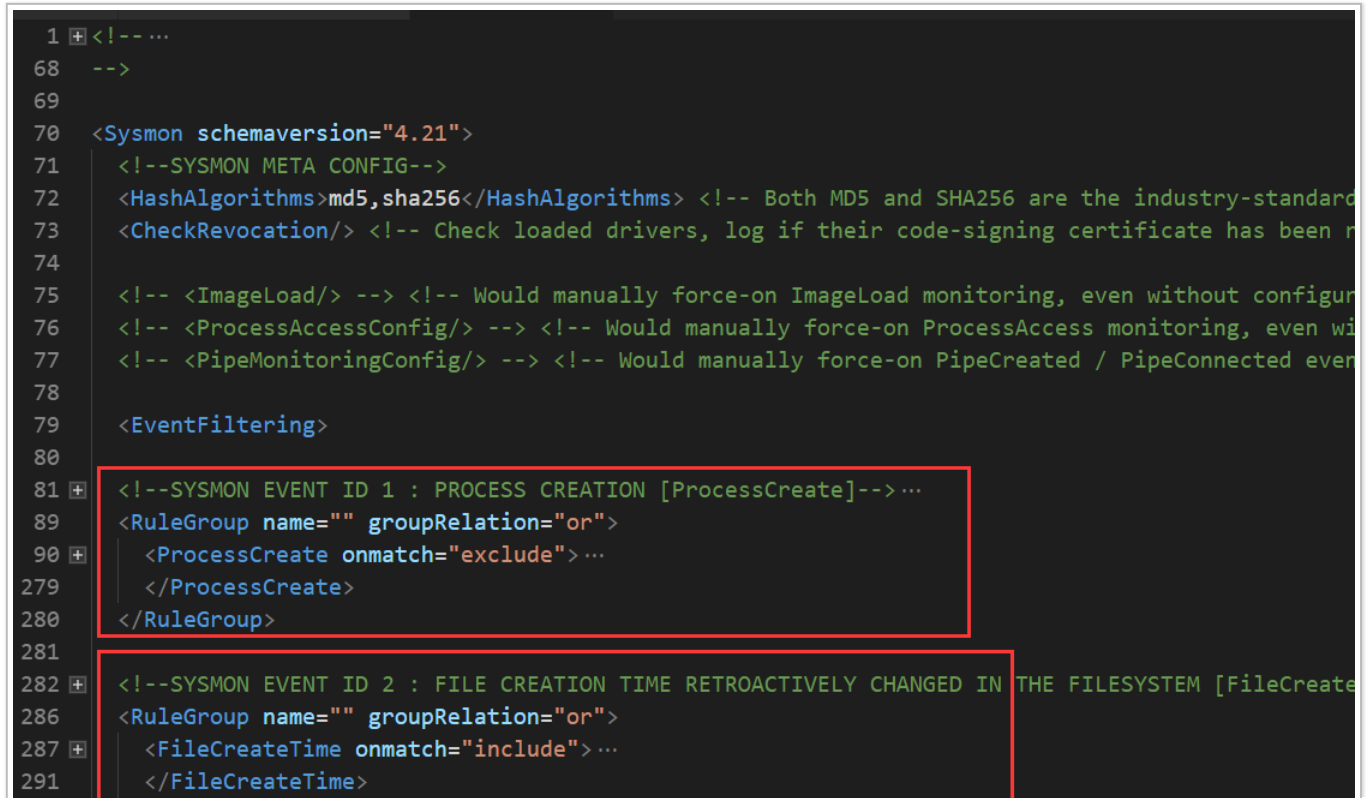
首先说明下有哪些事件过滤器。

其中只有 ID 4 和 16 无法过滤，其他可以根据 ID 标签来过滤。

ID	Tag
1 ProcessCreate	Process Create
2 FileCreateTime	File creation time
3 NetworkConnect	Network connection detected
4 n/a	Sysmon service state change (cannot be filtered)
5 ProcessTerminate	Process terminated
6 DriverLoad	Driver Loaded
7 ImageLoad	Image loaded
8 CreateRemoteThread	CreateRemoteThread detected
9 RawAccessRead	RawAccessRead detected
10 ProcessAccess	Process accessed
11 FileCreate	File created
12 RegistryEvent	Registry object added or deleted
13 RegistryEvent	Registry value set
14 RegistryEvent	Registry object renamed

15 FileCreateStreamHash	File stream created
16 n/a	Sysmon configuration change (cannot be filtered)
17 PipeEvent	Named pipe created
18 PipeEvent	Named pipe connected
19 WmiEvent	WMI filter
20 WmiEvent	WMI consumer
21 WmiEvent	WMI consumer filter
22 DNSQuery	DNS query

然后我们看下配置文件，这个配置文件基本每行都有做注释，很详细了。每个事件过滤器都写进去了，并提供推荐配置，只需在这基础上修改。先看一下结构。


1 <!--…
68 -->
69
70 <Sysmon schemaversion="4.21">
71 <!--SYSMON META CONFIG-->
72 <HashAlgorithms>md5,sha256</HashAlgorithms> <!-- Both MD5 and SHA256 are the industry-standard
73 <CheckRevocation/> <!-- Check loaded drivers, log if their code-signing certificate has been r
74
75 <!-- <ImageLoad/> --> <!-- Would manually force-on ImageLoad monitoring, even without configur
76 <!-- <ProcessAccessConfig/> --> <!-- Would manually force-on ProcessAccess monitoring, even wi
77 <!-- <PipeMonitoringConfig/> --> <!-- Would manually force-on PipeCreated / PipeConnected even
78
79 <EventFiltering>
80
81 <!--SYSMON EVENT ID 1 : PROCESS CREATION [ProcessCreate]-->…
89 <RuleGroup name="" groupRelation="or">
90 <ProcessCreate onmatch="exclude">…
279 </ProcessCreate>
280 </RuleGroup>
281
282 <!--SYSMON EVENT ID 2 : FILE CREATION TIME RETROACTIVELY CHANGED IN THE FILESYSTEM [FileCreate
286 <RuleGroup name="" groupRelation="or">
287 <FileCreateTime onmatch="include">…
291 </FileCreateTime>

```
292
293   <FileCreateTime onmatch="exclude">···
302   </FileCreateTime>
303 </RuleGroup>
304
305 <!--SYSMON EVENT ID 3 : NETWORK CONNECTION INITIATED [NetworkConnect]-->···
315 <RuleGroup name="" groupRelation="or">
316   <NetworkConnect onmatch="include">···
396   </NetworkConnect>
397
398   <NetworkConnect onmatch="exclude">···
408   </NetworkConnect>
```

所有规则都在过滤器 <EventFiltering> 标签里。

然后各类事件单独封装在各个 <RuleGroup> 里，也就是说根据上面支持的事件过滤器，一共有 15 个 < RuleGroup>，属性 groupRelation 都设置为 or，表明事件标签里同一规则的每个项之间是或的关系。

选择以下事件标签来讲解，标签名称为 FileCreateTime，表明过滤进程修改文件创建时间事件，属性 onmatch 为 include 表示匹配以下标签规则的才记录日志，exclude 值，表示不匹配以下标签规则的才记录日志，exclude 规则覆盖 include 规则。

事件标签内可以看到多个 Image 项，各项由于 groupRelation 设置为 or，各项之间是或的关系，看下图存在两个相同的事件标签，但一个是 include，一个是 exclude，相同事件标签之间是与的关系。按下面规则表示不匹配镜像路径为“OneDrive.exe”或“C:\Windows\system32\backgroundTaskHost.exe”或包含“setup”字符或…… 的，同时只匹配镜像路径以“C:\Users”开头或以“.exe”结尾等等的。name 属性表示规则名时，当匹配到其中某条规则日志会记录匹配到的规则名，condition 表示匹配方式，如”begin with”表示以该值为开头，”image”表示匹配镜像路径，默认为 is，精确匹配。

```
2 <!--SYSMON EVENT ID 2 : FILE CREATION TIME RETROACTIVELY CHANGED IN THE FILESYSTEM [FileCreateTime]
6 <RuleGroup name="" groupRelation="or">
```

```
    <FileCreateTime onmatch="include">
      <Image name="T1099" condition="begin with">C:\Users</Image> <!--Look for timestomping in user ar
      <TargetFilename name="T1099" condition="end with">.exe</TargetFilename> <!--Look for backdated e
      <Image name="T1099" condition="begin with">\Device\HarddiskVolumeShadowCopy</Image> <!--Nothing
    </FileCreateTime>

    <FileCreateTime onmatch="exclude">
      <Image condition="image">OneDrive.exe</Image> <!--OneDrive constantly changes file times-->
      <Image condition="image">C:\Windows\system32\backgroundTaskHost.exe</Image>
      <Image condition="contains">setup</Image> <!--Ignore setups-->
      <Image condition="contains">install</Image> <!--Ignore setups-->
      <Image condition="contains">Update\</Image> <!--Ignore setups-->
      <Image condition="end with">redist.exe</Image> <!--Ignore setups-->
      <Image condition="is">msiexec.exe</Image> <!--Ignore setups-->
      <Image condition="is">TrustedInstaller.exe</Image> <!--Ignore setups-->
    </FileCreateTime>
  </RuleGroup>
```

condition 的值有以下几种，还是比较好理解的。

Condition	Description
Is	Default, values are equals
is not	Values are different
Contains	The field contains this value
Excludes	The field does not contain this value
begin with	The field begins with this value
end with	The field ends with this value

end with	The field ends with this value
less than	Lexicographical comparison is less than zero
more than	Lexicographical comparison is more than zero
Image	Match an image path (full path or only image name). For example: lsass.exe will match c:\windows\system32\lsass.exe

每种事件有哪些项可以用来匹配过滤，如 NetworkConnect 的 Image 用来过滤镜像路径，这些可以参考最后 freebuf 的链接，这边主要讲自己的一些理解。

配置文件的结构分析差不多就这些，最后讲解下目前这个配置文件里各个事件规则的特点，方便大家进行适当修改调整。

1.ProcessCreate: 只有 exclude，不匹配系统自身默认会启动的一些服务程序，包括一些白名单程序启动。如果有额外的白名单程序不需要记录的可以添加进来。

2.FileCreateTime: 包括 include 和 exclude，主要记录 C:\Users 下或 exe 文件创建时间被修改，不记录一些安装程序修改记录，比如 setup/install 这一类名字的，当然可能会有病毒为这个名字，看具体情况调整。

3.NetworkConnect: 包括 include 和 exclude，这一块偏重于 include，只记录一些特殊的连接，因为电脑经常联网会有大量的应用连接，会导致大量无用日志记录，所以这边只设置了一些可疑程序以及可疑端口连接监控，像 powershell，cmd 等等，或者目标端口为 22,23,3389 这一类的。可能会有一些网络连接记录不到，如果需要记录一些其他的网络连接，该需要手动修改这部分内容。这一块强制不记录主要是微软的一些连接。

4.ProcessTerminate: 只有 include，记录设备程序被终止，或 C:\Users 下的程序被停止

5.DriverLoad: 只有 exclude，不记录签名包含 micrsoft、windows 或以 Intel 的驱动加载。

6.ImageLoad: 只有 include，但为空，表明不记录任何镜像加载。

7.CreateRemoteThread：只有 exclude，不记录一些系统程序的远程线程创建。

8.RawAccessRead: 只有 include，但为空，表明不记录该事件日志。

9.ProcessAccess: 只有 include，但为空，表明不记录该事件日志。

10.FileCreate: 包括 include 和 exclude，主要记录一些运行会触发病毒行为的程序，包括 exe、vbs、bat 等等，还有一些特殊路径下文件创建，如 startup 路径下有文件创建，就比较可疑。其他位置需要记录的，手动添加路径或文件名。不记录一些临时目录或一些白名单程序会创建文件的行为。

11.RegistryEvent: 包括 include 和 exclude，这两方面都有所平衡，一方面会记录病毒经常会访问的注册表路径，包括 run，services 一些启动项，RDP 等等，另一方面不记录一些系统本身会修改的注册表路径，不然会充斥太多无用日志。

12.FileCreateStreamHash: 只有 include，主要记录浏览器下载的一些可疑脚本文件。

13.PipeEvent: 只有 include，但为空，表明不记录该事件日志。

14.WmiEvent: 只有 include，但为空，表明不记录该事件日志。

15.DnsQuery: 只有 exclude，这块就是排除一些正常并且频繁发生的域名解析，避免干扰。如果只需要记录某个域名解析，其实可以单独设置 include 过滤。

写这篇文章的初衷，主要是最近工具发布了 dns 查询功能，想好好利用这个工具，并且在以往的使用中也发现这个工具蛮好用的，所以趁此机会，好好学一学这工具的使用，学习中发现在 github 上发现有人整理了这工具的配置文件，也刚好可以通过分析这个配置文件，更全面的了解这个工具，然后将自己学习理解的内容整理成笔记。当然我也只用到了其中一小部分的功能，后续还需要结合具体安全事件来更好理解使用工具。

https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon

https://www.freebuf.com/sectool/122779.html