

Kevin-Robertson / Inveigh

Public

Notifications

Fork 444

Star 2.5k

<> Code

Issues 19

Pull requests 1

Actions

Projects

Wiki

Security

Insights

Files

29d9e3c

Go to file

> .github

> Inveigh

> Listeners

> Protocols

> Sniffer

> Sockets

> Support

Arguments.cs

Control.cs

Output.cs

Shell.cs

FodyWeavers.xml

FodyWeavers.xsd

Inveigh.csproj

Program.cs

.gitattributes

.gitignore

Inveigh-Relay.ps1

Inveigh.ps1

Inveigh.psd1

Inveigh.psm1

Inveigh.sln

LICENSE

README.md

Inveigh / Inveigh / Support / Control.cs

Kevin-Robertson interval fix, DNS AAAA 28ffe89 · 2 years ago History

Code

Blame

613 lines (488 loc) · 25.3 KB

Raw

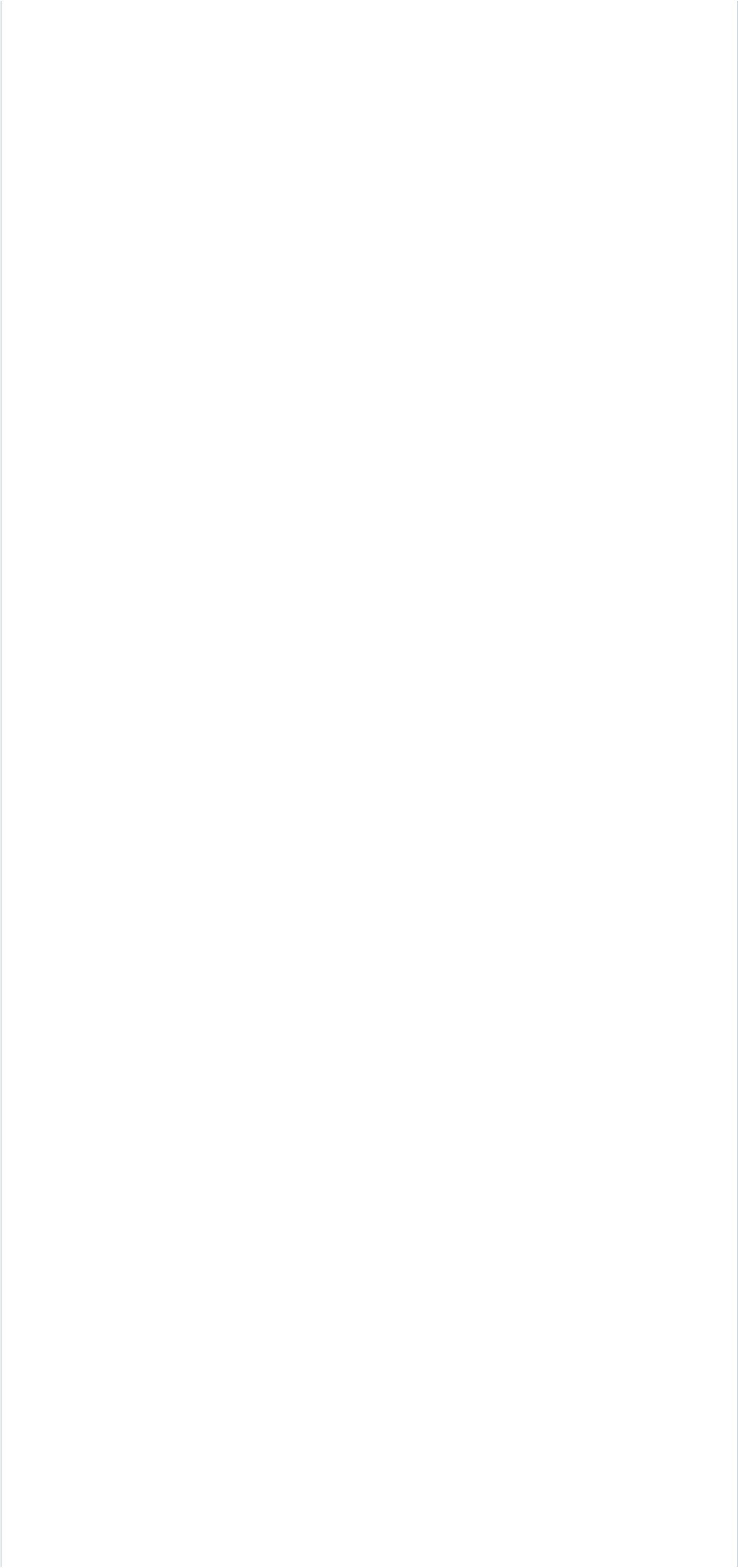
```
1      using System;
2      using System.IO;
3      using System.Linq;
4      using System.Threading;
5      using System.Diagnostics;
6      using System.Text.RegularExpressions;
7      using System.Net;
8      using System.Collections.Generic;
9      using System.Text;
10     using System.Globalization;
11
12     namespace Inveigh
13     {
14         class Control
15         {
16
17             public static void ControlLoop(string consoleLevel, int consoleQueueLimit, int
18             {
19                 Stopwatch stopwatchConsoleStatus = new Stopwatch();
20                 stopwatchConsoleStatus.Start();
21                 Stopwatch stopwatchRunTime = new Stopwatch();
22                 stopwatchRunTime.Start();
23                 bool isPromptRefresh = false;
24
25                 while (Program.isRunning)
26                 {
27
28                     if (Program.clearTextList.Count > Program.clearTextCount)
29                     {
30                         Program.clearTextCount = Program.clearTextList.Count;
31                         Program.isClearTextUpdated = true;
32                         isPromptRefresh = true;
33                     }
34
35                     if (Program.ntlmv1List.Count > Program.ntlmv1Count)
36                     {
37                         Program.ntlmv1Count = Program.ntlmv1List.Count;
38                         Program.isNTLMv1Updated = true;
39                         isPromptRefresh = true;
40                     }
41
42                     if (Program.ntlmv2List.Count > Program.ntlmv2Count)
43                     {
44                         Program.ntlmv2Count = Program.ntlmv2List.Count;
45                         Program.isNTLMv2Updated = true;
46                         isPromptRefresh = true;
47                     }
48
49                     IList<string> clearTextUnique = Shell.GetUnique(Program.clearTextList,
50
51                     if (clearTextUnique.Count > Program.clearTextUniqueCount)
52                     {
53                         Program.clearTextUniqueCount = clearTextUnique.Count;
54                         Program.isClearTextUniqueUpdated = true;
55                         isPromptRefresh = true;
56                     }
57                 }
58             }
59         }
60     }
```

Page 1 of 9

```
58         if (Program.ntlmv1UniqueList.Count > Program.ntlmv1UniqueCount)
59         {
60             Program.ntlmv1UniqueCount = Program.ntlmv1UniqueList.Count;
61             Program.isNTLMv1UniqueUpdated = true;
62             isPromptRefresh = true;
63         }
64
65         if (Program.ntlmv2UniqueList.Count > Program.ntlmv2UniqueCount)
66         {
67             Program.ntlmv2UniqueCount = Program.ntlmv2UniqueList.Count;
68             Program.isNTLMv2UniqueUpdated = true;
69             isPromptRefresh = true;
70         }
71
72         if (isPromptRefresh && !Program.enabledConsoleOutput)
73         {
74             Shell.RefreshCurrentLine();
75             isPromptRefresh = false;
76         }
77         else
78         {
79             isPromptRefresh = false;
80         }
81
82         if (consoleStatus > 0 && Program.enabledConsoleOutput && stopwatchConso
83         {
84             Shell.GetCleartextUnique("");
85             Shell.GetNTLMv1Unique("");
86             Shell.GetNTLMv1Usernames("");
87             Shell.GetNTLMv2Unique("");
88             Shell.GetNTLMv2Usernames("");
89             stopwatchConsoleStatus.Reset();
90             stopwatchConsoleStatus.Start();
91         }
92
93         if (runTime > 0 && Program.enabledConsoleOutput && stopwatchRunTime.Ela
94         {
95             Output.Queue(String.Format("[*] {0} Inveigh is exiting due to reach
96             StopInveigh());
97         }
98
99         if (runCount > 0 && Program.enabledConsoleOutput && (Program.ntlmv1List
100         {
101             Output.Queue(String.Format("[*] {0} Inveigh is exiting due to reach
102             StopInveigh());
103         }
104
105         try
106         {
107             Output.ProcessOutput();
108         }
109         catch (Exception ex)
110         {
111             Output.Queue(String.Format("[-] [{0}] Output error detected - {1}",
112             }
113
114         Thread.Sleep(5);
115     }
116
117 }
118
```










```
540         EnabledWebDAV = true,
541         IgnoreAgents = Program.argIgnoreAgents,
542         HTTPAuth = Program.argHTTPAuth,
543         WebDAVAuth = Program.argWebDAVAuth,
544         WPADAuth = Program.argWPADAuth,
545         HTTPRealm = Program.argHTTPRealm,
546         HTTPResponse = Program.argHTTPResponse,
547         WPADResponse = Program.argWPADResponse,
548         NetbiosDomain = Program.netbiosDomain,
549         ComputerName = Program.computerName,
550         DNSDomain = Program.dnsDomain
551     };
552
553     Thread httpv6ListenerThread = new Thread(() => httpv6Listen
554     httpv6ListenerThread.Start();
555 }
556
557 }
558
559 if (Program.enabledHTTPS)
560 {
561
562     foreach (string port in Program.argHTTPPorts)
563     {
564
565         HTTPListener httpsv6Listener = new HTTPListener
566         {
567             Challenge = Program.argChallenge,
568             Cert = Program.argCert,
569             CertPassword = Program.argCertPassword,
570             EnabledWebDAV = true,
571             IgnoreAgents = Program.argIgnoreAgents,
572             HTTPAuth = Program.argHTTPAuth,
573             WebDAVAuth = Program.argWebDAVAuth,
574             WPADAuth = Program.argWPADAuth,
575             HTTPRealm = Program.argHTTPRealm,
576             HTTPResponse = Program.argHTTPResponse,
577             WPADResponse = Program.argWPADResponse,
578             NetbiosDomain = Program.netbiosDomain,
579             ComputerName = Program.computerName.
```



```
-
580             DNSDomain = Program.dnsDomain
581         };
582
583         Thread httpsv6ListenerThread = new Thread(() => httpsv6List
584             httpsv6ListenerThread.Start();
585     }
586
587 }
588
589     if (Program.enabledICMPv6) // todo check linux
590     {
591         ICMPv6Socket icmpV6Socket = new ICMPv6Socket();
592         Thread icmpv6Thread = new Thread(() => icmpV6Socket.Start());
593         icmpv6Thread.Start();
594     }
595
596 }
597
598 }
599
600     Thread controlThread = new Thread(() => ControlLoop(Program.argConsole, Pro
601         controlThread.Start();
602
603     if (Program.enabledFileOutput)
604     {
605         Thread fileOutputThread = new Thread(() => Output.FileOutput());
606         fileOutputThread.Start();
607     }
608
609 }
610
611 }
612
613 }
```