Equation DSz

概述

作者根据EQGRP公开资料进行研究分析，研究相关工具的开发实现和攻击防御思路。

NSA 是美国网空对抗的主力军，分析NSA的C2平台对于进一步了解NSA的网空行为非常必要。

DSz是一个对WIndows系统进行渗透的C2平台，采用Python2和Java技术进行构建，整个平台采用模块化方式进行组合，配合脚本系统，可以方便的大范围监控目标计算机系统，实现数据的实时抓取，完成目标的实时监控。

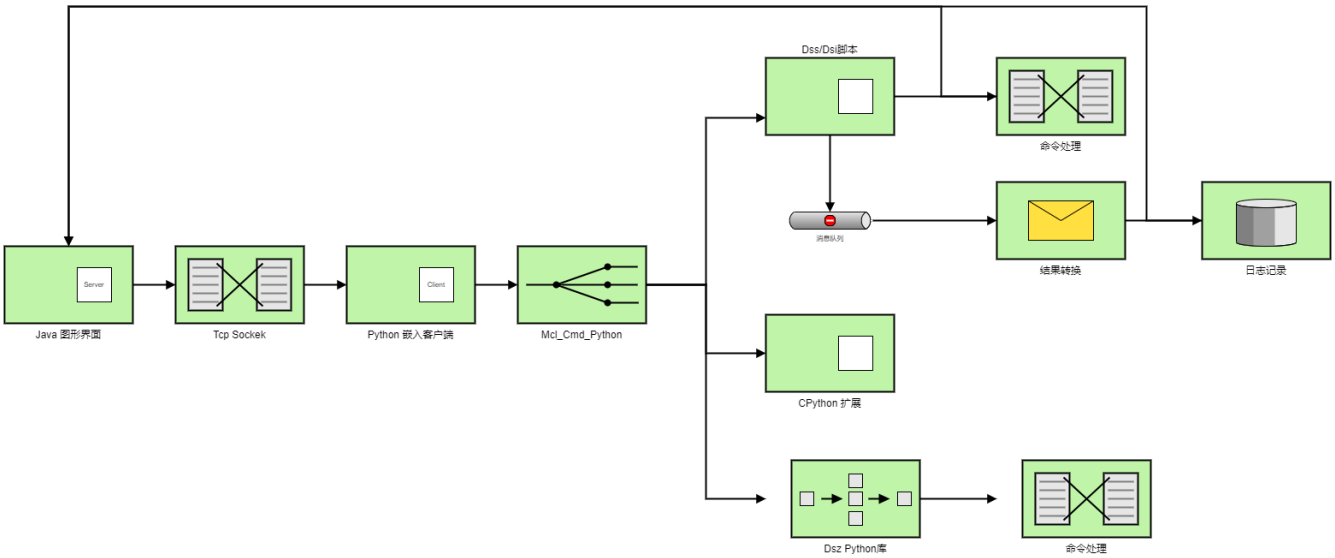
架构

为了分析DSz的功能和结构，重新构建DSz的最小运行构成，发现需要三个模块，DSz，GUI和Python。

```
Show-Tree D:\work\malware\bvp47\FB\Resources -depth 2
D:\work\malware\bvp47\FB\Resources
├──Dsz
│   ├──Aliases
│   ├──Commands
│   ├──CoreSchemas
│   ├──Data
│   ├──Databases
│   ├──Gui
│   ├──Modules
│   ├──Payloads
│   ├──PyLp
│   ├──PyScripts
│   ├──Scripts
│   ├──Tools
│   └──Version
├──ExternalLibraries
│   ├──i386-winnt-vc71
│   ├──i386-winnt-vc9
│   └──java-j2se_1.6-sun
├──Gui
│   ├──Config
│   ├──lib
│   └──Scripts
└──Python
```

```
    ├──Core
    └──Override
```

其中Python模块就是Python的标准嵌入式发布，GUI则是Java Swing图形界面代码，Dsz是命令执行的实现，ExternalLibrarie是Windows程序DszLpCore.exe依赖的库文件。
整个代码的架构如下所示：



Java的GUI启动一个Socket服务端，然后等待Python客户端来连接，连接后，循环读取用户输入，返回执行结果。

流程

Start

程序的启动依赖Java运行环境，程序木有自带，需要自己准备java8运行环境。

```
java -jar Start.jar
```

Start设置Theme，然后分析参数，最后启动，读取配置文件user.defaults, start.properties，生成一个自定义的URLClassLoader，然后执行ds.plugin.live.DSClientApp，Start类完成自己的使命，进入GUI界面。

GUI

```
Show-Tree D:\work\malware\bvp47\fuzzbunch\Resources\Gui -Depth 2
D:\work\malware\bvp47\fuzzbunch\Resources\Gui
├──Config
│   ├──Console
```

```
│   ├──FileBrowser
│   ├──Modules
│   ├──NetmapViewer
│   ├──RequestHandler
│   ├──ScreenShot
│   ├──ScriptEditor
│   ├──Shell
│   ├──TargetDetails
│   ├──TaskManager
│   └──Terminal
├──lib
│   └──java-j2se_1.6-sun
└──Scripts
    └──RequestHandler

dir lib\java-j2se_1.6-sun
----                 -------------         ------ ----
-a---            2022/4/4     9:07          32635 About.jar
-a---            2022/4/4     9:07          22789 CommandViewer.jar
-a---            2022/4/4     9:07          51135 ConnectionInfo.jar
-a---            2022/4/4     9:07         159023 Console.jar
-a---            2022/4/4     9:07          33458 DebugViewer.jar
-a---            2022/4/4     9:07         919541 FileBrowser.jar
-a---            2022/4/4     9:07          55538 GeneralUtilities.jar
-a---            2022/4/4     9:07         354198 GuiUtilities.jar
-a---            2022/4/4     9:07         670152 Icons.jar
-a---            2022/4/4     9:07         173848 Interface.jar
-a---            2022/4/4     9:07          19867 JavaLogViewer.jar
-a---            2022/4/4     9:07         304111 LogViewer.jar
-a---            2022/4/4     9:07          31837 Mirror.jar
-a---            2022/4/4     9:07          51673 Monitor.jar
-a---            2022/4/4     9:07         220228 NetmapViewer.jar
-a---            2022/4/4     9:07         205248 Notify.jar
-a---            2022/4/4     9:07         121630 RequestHandler.jar
-a---            2022/4/4     9:07         120835 ScreenShot.jar
-a---            2022/4/4     9:07         202200 ScriptEditor.jar
-a---            2022/4/4     9:07          20449 Shell.jar
-a---            2022/4/4     9:07          56176 SystemLogViewer.jar
-a---            2022/4/4     9:07          73062 TableUtilities.jar
-a---            2022/4/4     9:07         183139 TargetDetails.jar
-a---            2022/4/4     9:07         117103 TargetModel.jar
-a---            2022/4/4     9:07         174463 TaskManager.jar
-a---            2022/4/4     9:07          35906 Terminal.jar
-a---            2022/4/4     9:07         112220 TransferMonitor.jar
```

GUI包含众多的jar文件，每个文件完成独立的功能。
DSClientApp扫描插件列表，创建用户界面，调用启动函数。

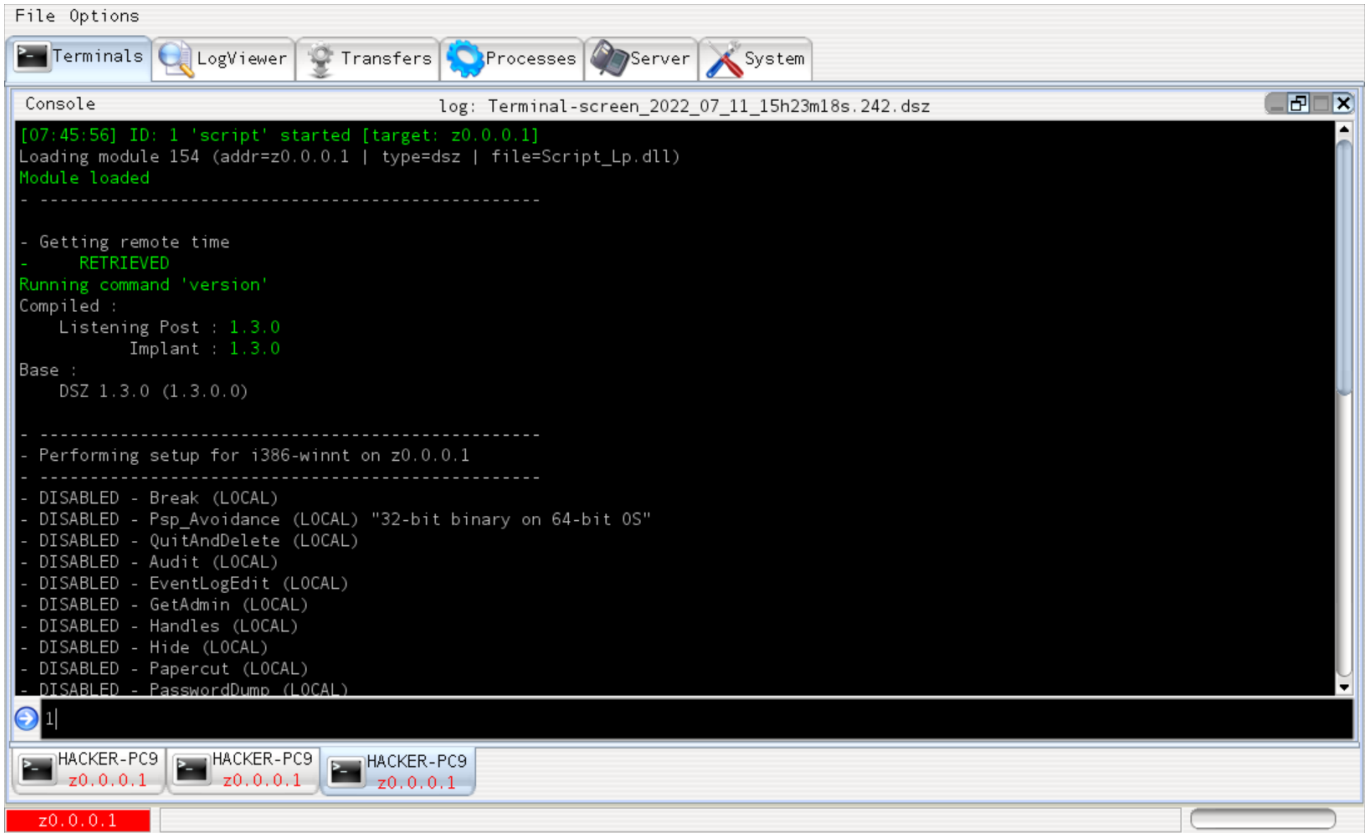
ImplantConsole

Dsz的插件启动时会执行ddb.dsz.plugin.appconsole.ImplantConsole，完成Python客户端的创建，从而实现界面操作和后台逻辑实现。

> 这里是一个反向控制，Python其实是服务端，Java才是客户端，控制反转的一个例子。

ImplantConsole 添加环境信息，创建DszLp.exe，然后启动。
DszLp.exe启动DszLpCore.exe，由后者完成Python部分的创建。

启动后，系统的界面部分如下：



启动Script_Lp.dll，这个dll用来实现dss脚本的解析。
在这个Terminal中可以执行dss脚本。

```
07:59:07>> script hello.dss
[07:59:07] ID: 85 'script' started [target: z0.0.0.1]
- ------------------------------------------------

- Hello world!
- HACKER-PC9
```

其中的脚本内容如下。

```
type .\hello.dss

@include "_Versions.dsi";

@echo off;

bool $rtn=true;
string $hostname;

echo "--------------------------------------------------\n";
echo "Hello world!";



GetEnv("_HOSTNAME", $hostname);
echo $hostname;


return $rtn;
```

到目前为止，系统启动完毕。

Console输出信息

```
[03:22:24] ID: 1 'script' started [target: z0.0.0.1]
Loading module 154 (addr=z0.0.0.1 | type=dsz | file=Script_Lp.dll)
Module loaded
- ------------------------------------------------

- Getting remote time
-     RETRIEVED
Running command 'version'
Compiled :
    Listening Post : 1.3.0
           Implant : 1.3.0
Base :
    DSZ 1.3.0 (1.3.0.0)

- ------------------------------------------------
- Performing setup for i386-winnt on z0.0.0.1
- ------------------------------------------------
- DISABLED - Break (LOCAL)
- DISABLED - Psp_Avoidance (LOCAL) "32-bit binary on 64-bit OS"
- DISABLED - QuitAndDelete (LOCAL)
- DISABLED - Audit (LOCAL)
```

```
- DISABLED - EventLogEdit (LOCAL)
- DISABLED - GetAdmin (LOCAL)
- DISABLED - Handles (LOCAL)
- DISABLED - Hide (LOCAL)
- DISABLED - Papercut (LOCAL)
- DISABLED - PasswordDump (LOCAL)
- DISABLED - Portmap (LOCAL)
- DISABLED - ProcessModify (LOCAL)
- DISABLED - ProcessOptions (LOCAL)
- DISABLED - RunAsChild (LOCAL)
- DISABLED - RunAsSystem (LOCAL)
- DISABLED - Shutdown (LOCAL)
- ------------------------------------------------

- Registering Mcl_NtElevation options
-     SUCCESS
- Registering Mcl_NtNativeApi options
-     SUCCESS
- Setting Mcl_NtNativeApi Type
-     WIN32
- Registering Mcl_NtMemory options
-     SUCCESS
- Setting Mcl_NtMemory Type
-     DrNi
- Registering Mcl_ThreadInject options
-     SUCCESS
- Setting Mcl_ThreadInject Type
-     DrNi
- Getting host information
-     RETRIEVED
- Getting OS GUID information
-     RETRIEVED
- Storing host information
-     STORED
```

整个模块是由脚本来控制启动。
Dsz的模块版本是1.3.0，包含的子模块LP，Implant的版本都是1.3.0。

根据运行环境禁止了部分功能，注册成功了部分功能。

在console输入help命令，可以显示全部的命令。

```
08:19:27>> help
[08:19:27] ID: 85 'help' started [target: z0.0.0.1]
Prefixes:
```

```
  async             background        disablewow64      foreground        guiflag
  local             log               src               stopaliasing      task
  dst               user              wait              xml
nocharescapes
  framework         disablepre        disablepost

Commands:

  activedirectory     activity            addresses           aliases
  arp                 audit               available           banner
  break               cd                  commands            copy
  cprpc               currentusers        database            delete
  devicequery         dir                 diskspace           dllload
  dns                 domaincontroller    drivers             drives
  environment         eventlogclear       eventlogedit        eventlogfilter
  eventlogquery       fileattributes      filetype            firewall
  freeplugin          frzaddress          frzlinks            frzroutes
  frzsecassocs        frztimeouts         generatedata        get
  getadmin            grep                groups              gui
  handles             help                hide                ifconfig
  keepalive           kill                language            ldap
  library             loadplugin          logedit             lpdirectory
  lpgetenv            lpsetenv            matchfiletimes      memory
  mkdir               moduletoggle        move
nameserverlookup
  netbios             netconnections      netmap              objects
  packages            packetredirect      papercut            passworddump
  performance         permissions         ping                plugins
  policy              portmap             processes           processinfo
  processmodify       processoptions      processsuspend      put
  pwd                 python              quitanddelete       redirect
  registryadd         registrydelete      registryhive        registryquery
  rmdir               route               run                 runaschild
  scheduler           script              serialredirect      services
  shutdown            sidlookup           stop                strings
  systempaths         systemversion       throttle            time
  traceroute          uptime              users               version
  warn                whoami              windows             wrappers
  xmlparser

- Loaded commands have a '*' preceeding the command name

For additional information try: help <command>


    Command completed successfully
```

这些命令对应的dll文件如下。

```
Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a---          2022/4/4     9:07          37888 ActiveDirectory_Target.dll
-a---          2022/4/4     9:07          26112 Activity_Target.dll
-a---          2022/4/4     9:07          31744 Arp_Target.dll
-a---          2022/4/4     9:07          28160 Banner_Target.dll
-a---          2022/4/4     9:07          27648 Cd_Target.dll
-a---          2022/4/4     9:07          25088 Copy_Target.dll
-a---          2022/4/4     9:07          24064 CurrentUsers_Target.dll
-a---          2022/4/4     9:07         365056 Database_Lp.dll
-a---          2022/4/4     9:07          48128 Delete_Target.dll
-a---          2022/4/4     9:07          27648 DeviceQuery_Target.dll
-a---          2022/4/4     9:07          51200 Dir_Target.dll
-a---          2022/4/4     9:07          25600 DiskSpace_Target.dll
-a---          2022/4/4     9:07          32256 DllLoad_Target.dll
-a---          2022/4/4     9:07          33792 Dns_Target.dll
-a---          2022/4/4     9:07          27648 DomainController_Target.dll
-a---          2022/4/4     9:07          34304 Drivers_Target.dll
-a---          2022/4/4     9:07          23552 Drives_Target.dll
-a---          2022/4/4     9:07         844800 Dsz_Error.dll
-a---          2022/4/4     9:07          27648 Environment_Target.dll
-a---          2022/4/4     9:07          27136 EventLogClear_Target.dll
-a---          2022/4/4     9:07          46080 EventLogQuery_Target.dll
-a---          2022/4/4     9:07          34816 FileAttributes_Target.dll
-a---          2022/4/4     9:07          51200 FileType_Lp.dll
-a---          2022/4/4     9:07          46592 Firewall_Target.dll
-a---          2022/4/4     9:07          60416 Friezeramp_Data.dll
-a---          2022/4/4     9:07          61952 Friezeramp_Lp.dll
-a---          2022/4/4     9:07          44544 Get_Data.dll
-a---          2022/4/4     9:07          90112 Get_Lp.dll
-a---          2022/4/4     9:07          58880 Get_Target.dll
-a---          2022/4/4     9:07          36864 Grep_Target.dll
-a---          2022/4/4     9:07          26624 Groups_Target.dll
-a---          2022/4/4     9:07          32256 Ifconfig_Target.dll
-a---          2022/4/4     9:07          21504 Kill_Target.dll
-a---          2022/4/4     9:07          27648 Language_Target.dll
-a---          2022/4/4     9:07          31232 Ldap_Target.dll
-a---          2022/4/4     9:07          27648 LogEdit_Target.dll
-a---          2022/4/4     9:07          41984 MatchFileTimes_Target.dll
-a---          2022/4/4     9:07           9728 Mcl_NtElevation_Fail.dll
-a---          2022/4/4     9:07           9728 Mcl_NtMemory_Fail.dll
-a---          2022/4/4     9:07          23040 Mcl_NtMemory_Std.dll
-a---          2022/4/4     9:07           9728 Mcl_NtNativeApi_Fail.dll
-a---          2022/4/4     9:07          20480 Mcl_NtNativeApi_Win32.dll
-a---          2022/4/4     9:07           9728 Mcl_ThreadInject_Fail.dll
-a---          2022/4/4     9:07          22016 Memory_Target.dll
```

```
-a---           2022/4/4     9:07          24576 Mkdir_Target.dll
-a---           2022/4/4     9:07          25088 Move_Target.dll
-a---           2022/4/4     9:07          31232 NameServerLookup_Target.dll
-a---           2022/4/4     9:07          28672 NetBios_Target.dll
-a---           2022/4/4     9:07          37376 NetConnections_Target.dll
-a---           2022/4/4     9:07          34816 Netmap_Target.dll
-a---           2022/4/4     9:07          27648 Objects_Target.dll
-a---           2022/4/4     9:07          29696 Packages_Target.dll
-a---           2022/4/4     9:07          27648 PacketRedirect_Target.dll
-a---           2022/4/4     9:07          67072 Papercut_Data.dll
-a---           2022/4/4     9:07          55296 Papercut_Lp.dll
-a---           2022/4/4     9:07          43008 Papercut_Target.dll
-a---           2022/4/4     9:07          31232 Performance_Target.dll
-a---           2022/4/4     9:07          33792 Permissions_Target.dll
-a---           2022/4/4     9:07          35328 Ping_Target.dll
-a---           2022/4/4     9:07          50688 Processes_Target.dll
-a---           2022/4/4     9:07          50176 ProcessInfo_Target.dll
-a---           2022/4/4     9:07          44032 Put_Target.dll
-a---           2022/4/4     9:07          95744 Redirect_Lp.dll
-a---           2022/4/4     9:07          49664 Redirect_Target.dll
-a---           2022/4/4     9:07          36352 RegistryHive_Target.dll
-a---           2022/4/4     9:07          37376 RegistryKeys_Target.dll
-a---           2022/4/4     9:07          43520 RegistryQuery_Target.dll
-a---           2022/4/4     9:07          24576 Rmdir_Target.dll
-a---           2022/4/4     9:07          31744 Route_Target.dll
-a---           2022/4/4     9:07          70656 Run_Data.dll
-a---           2022/4/4     9:07          56832 Run_Lp.dll
-a---           2022/4/4     9:07          37888 Run_Target.dll
-a---           2022/4/4     9:07          67072 Scheduler_Target.dll
-a---           2022/4/4     9:07         342016 Script_Lp.dll
-a---           2022/4/4     9:07          67072 SerialRedirect_Lp.dll
-a---           2022/4/4     9:07          33280 SerialRedirect_Target.dll
-a---           2022/4/4     9:07          26112 Services_Target.dll
-a---           2022/4/4     9:07          22016 Shutdown_Target.dll
-a---           2022/4/4     9:07          27648 SidLookup_Target.dll
-a---           2022/4/4     9:07          28672 Strings_Target.dll
-a---           2022/4/4     9:07          22528 SystemPaths_Target.dll
-a---           2022/4/4     9:07          28672 SystemVersion_Target.dll
-a---           2022/4/4     9:07          47104 Time_Data.dll
-a---           2022/4/4     9:07          47616 Time_Lp.dll
-a---           2022/4/4     9:07          44544 Time_Target.dll
-a---           2022/4/4     9:07          43520 Traceroute_Target.dll
-a---           2022/4/4     9:07          27136 UpTime_Target.dll
-a---           2022/4/4     9:07          28160 Users_Target.dll
-a---           2022/4/4     9:07          46080 Windows_Target.dll
-a---           2022/4/4     9:07          45568 XmlParser_Lp.dll
```

这些DLL会发送到目标机，然后执行。

关键技术点

模块结构

以Dsz为例，看一下模块的构成。

```
Show-Tree . -Depth 2
D:\work\malware\bvp47\FB\Resources\Dsz
├──Aliases 命令描述
├──Commands 命令的输入输出参数
│   ├──CommandLine 输入参数
│   ├──Display 输出格式化xsl
│   └──Storage 持久化xsl
├──CoreSchemas 模块描述
├──Data 辅助数据
│   ├──disable
│   └──windows
├──Databases sqlite3 数据文件
├──Gui java代码，扩展GUI
│   ├──Config
│   └──lib
├──Modules 模块的扩展实现,采用C++开发对应dll
│   ├──Descriptions
│   ├──Files
│   ├──Files-dsz
│   └──Techniques
├──Payloads 投递到目标机的Payload, Dsz_Implant_Pc.dll
│   ├──Descriptions
│   └──Files
├──PyLp 数据转换xml
│   └──DataHandlers
├──PyScripts python脚本，用来扩展功能
│   ├──DataHandlers
│   ├──HelperScripts
│   ├──History
│   ├──Lib
│   ├──ModuleToggle
│   ├──Peel
│   ├──Policy
│   ├──Screenshot
│   └──Tasking
├──Scripts dss/dsi脚本，用来自动化操作
│   ├──Connected
│   ├──Flav
│   ├──HelperScripts
│   ├──Include
```

```
│   ├──PSP
│   └──RequestHandler
├──Tools 攻击jar
│   └──java-j2se_1.5
└──Version 版本信息xml
```

从模块的组成来看，使用的技术比较成熟，结构复杂，应该是开发了很长时间，有不少积累。对开发人员的要求比较高，需要掌握Java, C++, Python，定制脚本的开发能力。

Python 嵌入式开发

Python可以作为DLL嵌入到C++程序中，这也是Dsz的开发方式。
DszLpCore.exe嵌入了Python，标准的Python2.7的嵌入版本。只添加了一个库multiprocessing。

```
Show-Tree -Depth 3 .
D:\work\malware\bvp47\fuzzbunch\Resources\Python
├──Core
│   ├──DLLs
│   └──Lib
│       ├──bsddb
│       ├──compiler
│       ├──ctypes
│       ├──curses
│       ├──distutils
│       ├──email
│       ├──encodings
│       ├──hotshot
│       ├──idlelib
│       ├──importlib
│       ├──json
│       ├──lib-tk
│       ├──lib2to3
│       ├──logging
│       ├──msilib
│       ├──multiprocessing
│       ├──pydoc_data
│       ├──site-packages
│       ├──sqlite3
│       ├──unittest
│       ├──wsgiref
│       └──xml
└──Override
    └──Lib
        └──multiprocessing
```

在Dsz的Pyscript目录下。Python脚本的组成如下。

```
 Show-Tree -Depth 3 .
D:\work\malware\bvp47\fuzzbunch\Resources\Dsz\PyScripts
├──DataHandlers 数据格式转换
├──HelperScripts 辅助脚本
├──History 对payload进行转换
├──Lib
│   ├──dsz dsz的库文件
│   │   ├──data
│   │   ├──file
│   │   ├──lp
│   │   ├──mca
│   │   ├──mca_dsz
│   │   ├──mcf
│   │   ├──menu
│   │   ├──operational
│   │   ├──path
│   │   ├──payload
│   │   ├──process
│   │   ├──ui
│   │   ├──user
│   │   ├──version
│   │   └──windows
│   ├──gui GUI辅助脚本
│   │   └──networkserver
│   ├──mcl 利用Payload对目标进行操作
│   │   ├──data
│   │   ├──elevation
│   │   ├──framework
│   │   ├──hiding
│   │   ├──imports
│   │   ├──injection
│   │   ├──lp
│   │   ├──memory
│   │   ├──object
│   │   ├──os
│   │   ├──privilege
│   │   ├──status
│   │   ├──target
│   │   ├──tasking
│   │   └──tools
│   ├──mcl_platform mcl的平台部分
│   │   ├──data
│   │   ├──tasking
│   │   └──tools
│   └──policy 目标平台的权限处理
├──ModuleToggle 执行开关操作
```

```
├──Peel 检查提权信息
├──Policy 检查目标权限信息
├──Screenshot 录屏
└──Tasking Mcl_Cmd 对应的执行脚本，与DLL配合使用
```

这里有大量的dll和对应的Python脚本，是这个C2的核心模块。利用C++实现了部分核心功能，提高了运行速度。

dss脚本

dss脚本对应的命令是script，下面看看系统自动的部分dss脚本，学习一下。
选择查看_GetTimeInfo.dss，因为时间信息最简单，不涉及负责的逻辑。

```
@include "_Arrays.dsi";
@include "_Versions.dsi";

@echo off;

bool $rtn=true;

echo "-------------------------------------------------\n";
echo "Getting remote time";

@record on;
if (`time`)
{
        string $bias;
        if (!GetCmdData("TimeItem::Bias", $bias) || !defined($bias))
        {
                echo("    FAILED (bias not found)", WARNING);
                $rtn = false;
        }
        else if (!SetEnv("_TIME_BIAS", $bias))
        {
                echo("    FAILED (unable to set environment)", WARNING);
                $rtn = false;
        }
        else
        {
                echo("    RETRIEVED", GOOD);
        }
}
else
{
        echo("    FAILED", error);
```

```
        $rtn = false;
}
@record off;

if (($rtn == false) && _IsUnix())
{
        echo "Getting remote time using alternate method";

        string $time;

        @record on;
        if (!`run -command "date -R" -redirect`)
        {
                echo("    FAILED (date command failed)", error);
                $rtn = false;
        }
        else if (!GetCmdData("ProcessOutput::Output", $time) ||
!defined($time))
        {
                echo("    FAILED (failed to get date output)", error);
                $rtn = false;
        }
        else
        {
                string $tParts;
                if (RegExMatch("[0-9]{2}:[0-9]{2}:[0-9]{2} ([-+]{1})([0-9]
{2})([0-9]{2})", $time, $tParts) && (sizeof($tParts) == 3))
                {
                        if ((<int>$tParts[1] == 0) && (<int>$tParts[2] == 0))
                        {
                                # +0000
                                $tParts[0] = "";
                        }
                        else if ($tParts[0] == "+")
                        {
                                $tParts[0] = "-";
                        }
                        else
                        {
                                $tParts[0] = "";
                        }

                        string $bias = "$tParts[0]$tParts[1]:$tParts[2]";
                        if (!SetEnv("_TIME_BIAS", $bias))
                        {
                                echo("    FAILED (unable to set
environment)", WARNING);
                                $rtn = false;
                        }
```

```
                        else
                        {
                                echo("    PASSED", GOOD);
                        }
                }
                else
                {
                        echo("    FAILED (date doesn't match expected
format)", error);
                        $rtn = false;
                }
        }
}

return $rtn;
```

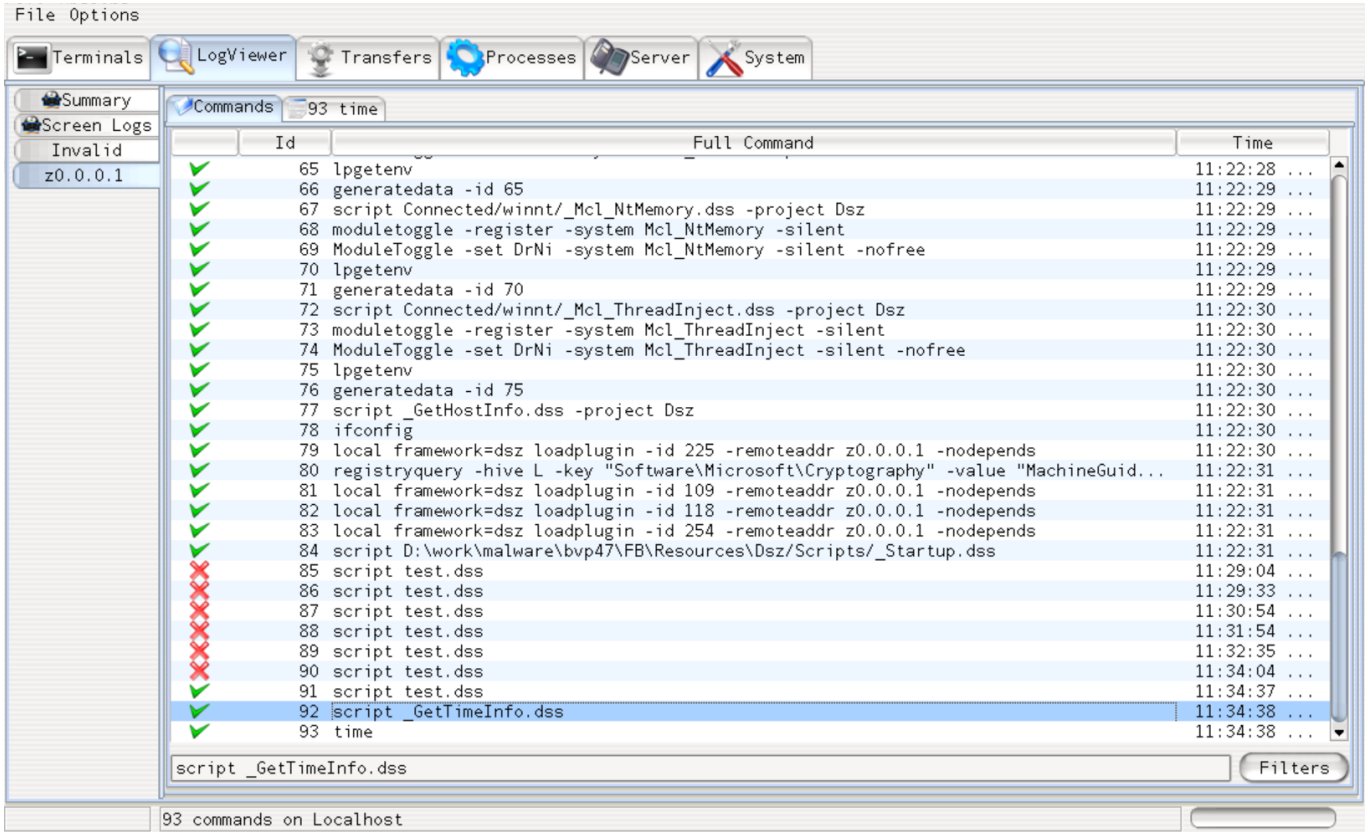
编写一个测试脚本test.dss。

```
@echo off;

bool $rtn=true;

`script _GetTimeInfo.dss`;


return $rtn;
```

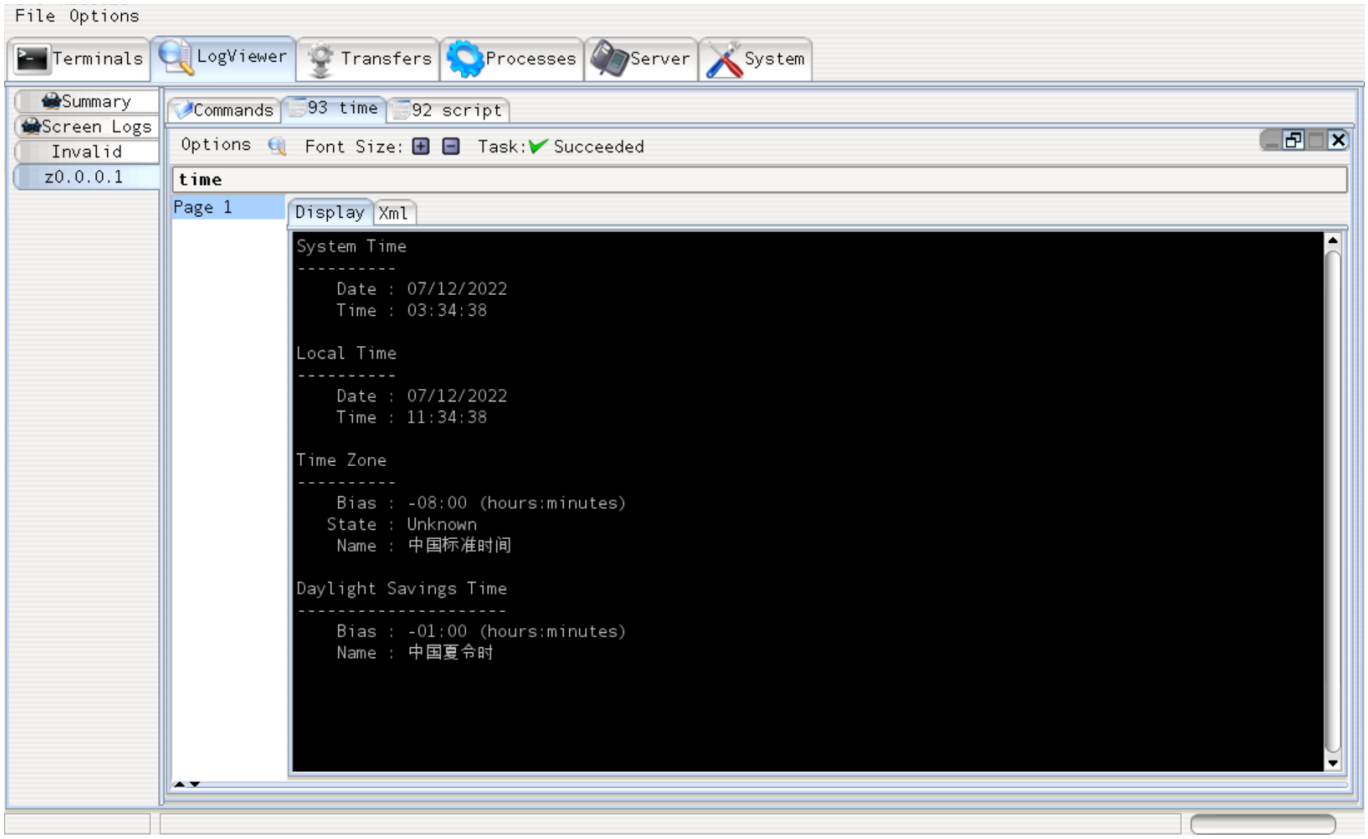
在Console中执行该脚本。

```
03:34:37>> script test.dss
[03:34:38] ID: 91 'script' started [target: z0.0.0.1]
- ------------------------------------------------

- Getting remote time
-     RETRIEVED
```

提示拿到了时间信息，查看日志窗口。



双击日志，会显示详细的信息。



显示了命令执行时的时间信息。

查看一下xml。

```
<?xml version='1.1' encoding='UTF-8' ?>
<DataLog xmlns='urn:mca:db00db84-8b5b-2141-a632b5980175d3c6'
xmlns:MCA='urn:mca:db00db84-8b5b-2141-a632b5980175d3c6'>
<Instance connectionId='cac09de4-15ba-5943-a8813b14019164c8'
coreVersion='1.3.0' id='d5cc687b-6efc-f94b-ba8df62509477f04'
lptimestamp='2022-07-12T03:34:38'/>
<Target addr='z0.0.0.1' arch='x64' cLibMajor='9' cLibMinor='0'
cLibRevision='0' compiledArch='i386' compiledPlatform='winnt'
majorVersion='6' minorVersion='2' otherVersion='0' pid='29776'
platform='winnt' taskId='a8dd585c-3fa8-5443-a9d59c9a245c690b'
tzOffset='-08:00'/>
<Command lptimestamp='2022-07-12T03:34:38' name='time'>
<!-- FullCommand: time -->
  <SourceAddress>z0.0.0.1</SourceAddress>
  <ResourceDir>Dsz</ResourceDir>
  <DisplayTransform>time_display.xsl</DisplayTransform>
  <StorageTransform>time_storage.xsl</StorageTransform>
  <Flags/>
</Command>
<CommandTasking>
<TaskingInfo lptimestamp='2022-07-12T03:34:38'>
  <CommandTarget type='local'/>
</TaskingInfo>
</CommandTasking>
<Success lptimestamp='2022-07-12T03:34:38'/>
</DataLog>
------------------------------------------------------------
<?xml version='1.1' encoding='UTF-8' ?>
<DataLog xmlns='urn:mca:db00db84-8b5b-2141-a632b5980175d3c6'
xmlns:MCA='urn:mca:db00db84-8b5b-2141-a632b5980175d3c6'>
<Instance connectionId='cac09de4-15ba-5943-a8813b14019164c8'
coreVersion='1.3.0' id='d5cc687b-6efc-f94b-ba8df62509477f04'
lptimestamp='2022-07-12T03:34:38'/>
<Target addr='z0.0.0.1' messageType='25020' taskId='a8dd585c-3fa8-5443-
a9d59c9a245c690b' tzOffset='-08:00'/>
<CommandData>
<Time dataTimestamp='2022-07-12T03:34:38' lptimestamp='2022-07-12T03:34:38'>
  <LocalTime type='remotelocal'>2022-07-12T11:34:38.180300200</LocalTime>
  <LocalTimeSeconds>1657625678</LocalTimeSeconds>
  <SystemTime type='remotegmt'>2022-07-12T03:34:38.180300200</SystemTime>
  <SystemTimeSeconds>1657596878</SystemTimeSeconds>
  <TimeZone>
    <Bias type='delta' negative='true'>P0DT08H00M00.000000000S</Bias>
    <CurrentState>Unknown</CurrentState>
  </TimeZone>
  <DaylightSavingsTime>
    <Standard>
      <Name>中国标准时间</Name>
```

```
        <Bias type='delta'>P0DT00H00M00.000000000S</Bias>
      </Standard>
      <Daylight>
        <Name>中国夏令时</Name>
        <Bias type='delta' negative='true'>P0DT01H00M00.000000000S</Bias>
      </Daylight>
    </DaylightSavingsTime>
  </Time>
  </CommandData>
  </DataLog>
  ------------------------------------------------------------
```

这个xml，更加详细的记录了命令的执行过程和结果。

总结

Dsz作为一个图形C2系统，方便了用户操作，降低了学习成本，利用了已有的Python程序，集成了脚本支持，是NSA进行大范围，大批量用户数据收集的一个优秀平台。

为了进一步分析其Loader，Payload的加载机制，需要分析其PC模块。

参考

1.