

**XÜSUSİ DÖVLƏT MÜHAFİZƏ XİDMƏTİ**  
**XÜSUSİ RABİTƏ VƏ İNFORMASIYA TƏHLÜKƏSİZLİYİ DÖVLƏT AGENTLİYİ**  
**KOMPÜTER İNSİDENTLƏRİNƏ QARŞI MÜBARİZƏ MƏRKƏZİ.**



**Malware Analysis Kit (MAK)**  
**Zərərverici Analizi Alətlər Toplusu**

Sistemin yaradılmasının əsas məqsədi  
Sistem haqqında ümumi informasiya  
Sistemin yüklənməsi və sazlamalar  
İstifadəçi təlimatları

## ÖN SÖZ

*Texnologiyaların yüksək sürətlə inkişafı həyatımıza müsbət təsiri ilə yanaşı bəzi mənfi cəhətləridə özüylə bərabər gətirir.*

*Bu mövzuyla əlaqədar nəzərdə tutduğumuz mənfi cəhətlər **Zərərli** proqram təminatlarıdır . (bundan sonra Malware)*

*Belə ki, bu tip proqram təminatlarını yazan şəxslərin əsas hədəfi texnologiya və iqtisadiyyatın inkişaf etdiyi ölkələr olur. Səbəbi isə bu ölkələrdə texnologiyanın inkişafı ilə insanlar artıq bir çox əməliyyatları internet üzərindən etməkləri onları birbaşa olaraq potensial hədəfə çevirir.*

*Və beləliklə “Malware” artıq hədəf istifadəçi ilə kibercinayətkar arasında körpü rolunu oynayır.*

*İstifadəçi yoluxdurulur və həmin istifadəçinin şəxsi kompüterindən alınan məlumatlar əsasında istifadəçinin şifrələri, bank əməliyyatları və sair məlumatları cinayətkar tərəfindən istifadə olunur.*

*KIMM Malware Lab-ın əsas məqsədi ölkəmizi hədəf alan bu tip proqram təminatlarının analiz edilməsi, onların müşahidə edilməsi, istifadəçilərin maarifləndirilməsi və bu sahənin inkişafına dəstək verməkdir.*

*Ümumilikdə **Malware Lab** - özündə bir sıra xidmətləri cəmləyən proqram təminatları ilə istifadəçilərə xidmət edəcək. Beləki, xidmətin birinci hissəsi zərərverici kodların analizi sahəsində professional məşğul olmaq istəyənlər üçün əlverişli analiz alətlərini istifadəyə vermək olacaqsə, ikinci hissədə sadə istifadəçilərin də yararlanma biləcəyi sadə interfeysli multi-funksional (virus aşkarlama motoru-antivirus, proses analiz modulu, trafik analiz modulu, zərərverici raporlama -malware reporting, məlumatlandırma modulu – CERT news və s.) analiz programından ibarət olacaq.*

## MAK nədir?

MAK - Windows Əməliyyat sistemində zərərvericiləri python interpreteri vasitəsi ilə “command line” rejimdə analiz etməyə və zərərvericilər haqqında informasiya yığmaq məqsədi ilə hazırlanıb. MAK məhz bu sahəyə dəstək vermək və bu sahənin inkişafı üçün yaradılmış bir sistem və alətlər toplusudur. MAK həmçinin növbəti layihə olan MAP-ın virus axtarış mühərrikində də “user firendly” interfeyslə istifadəçilə olunacaq.

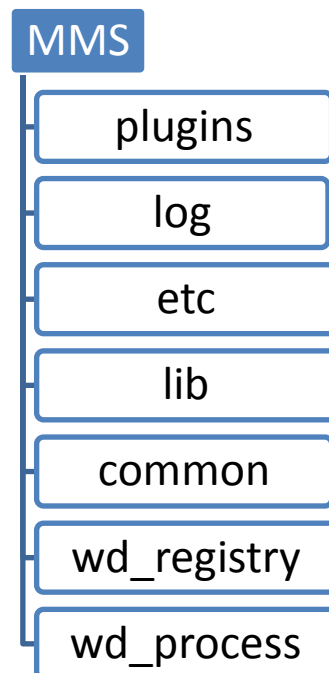
Əsas hissələri Python proqramlaşdırma dilində yazılmışdır. Bundan əlavə olaraq bəzi alt modullar üçün C proqramlaşdırma dilindən istifadə edilmişdir.

- Sistem bu sahədə müəyyən alt biliklərə sahib Malware analitikləri üçün nəzərdə tutulmuşdur.

Buna baxmayaraq Python dili haqqında müəyyən biliyə malik insanlar da sistemdən istifadə edə bilər.

MAK əsasən sistemdə müşahidələr aparmaq üçün nəzərdə tutulmuşdur. Lakin bundan əlavə olaraq istifadəçilər özləridə proseslərə müdaxilə edib onlar haqda məlumat toplaya bilərlər. Lakin bu tip müdaxilələr tam manual həyata keçirilməlidir. Səbəbi istifadəçilərin şüurlu bir şəkildə nəyi necə etdiklərini başa düşmələridir.

**MAK -ın əsas qovluq quruluşu aşağıdakı sxemdə göstərilmişdir.**

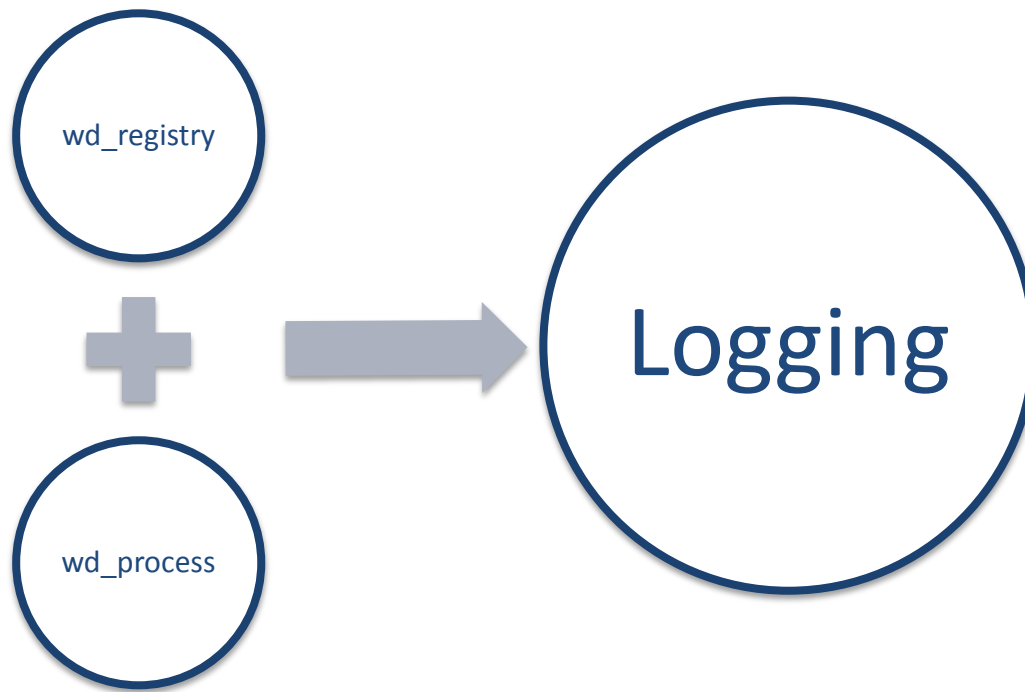


## Watchdog

1. wd\_registry(Registry dəyişikliklərini qeydə alır)
2. wd\_process(Yeni yaradılan prosesləri qeydə alır)

Bu modullar əsasən istifadəçiyə məlumat vermək üçün nəzərdə tutulmuşdur.

İşləmə qaydası aşağıda kışxemdə.



Gördüyünüz kimi qeydə alınan dəyişikliklər loqlaşdırılır.

Sonradan bu logları 8080 portdan paylaşa da bilərsiniz. (Sharing)

Bunları modullara aid müvafiq bölmələrdə tapa bilərsiniz.

Bundan əlavə olaraq plugin bölməsində pluginlərə rast gələ bilərsiniz həmin pluginləri əsas modula əlavə edib istifadə edə bilərsiniz.

- ✓ MSS istifadəçilərdən də python dilində yazdıqları pluginləri qəbul edir və onları sistemə əlavə edir.

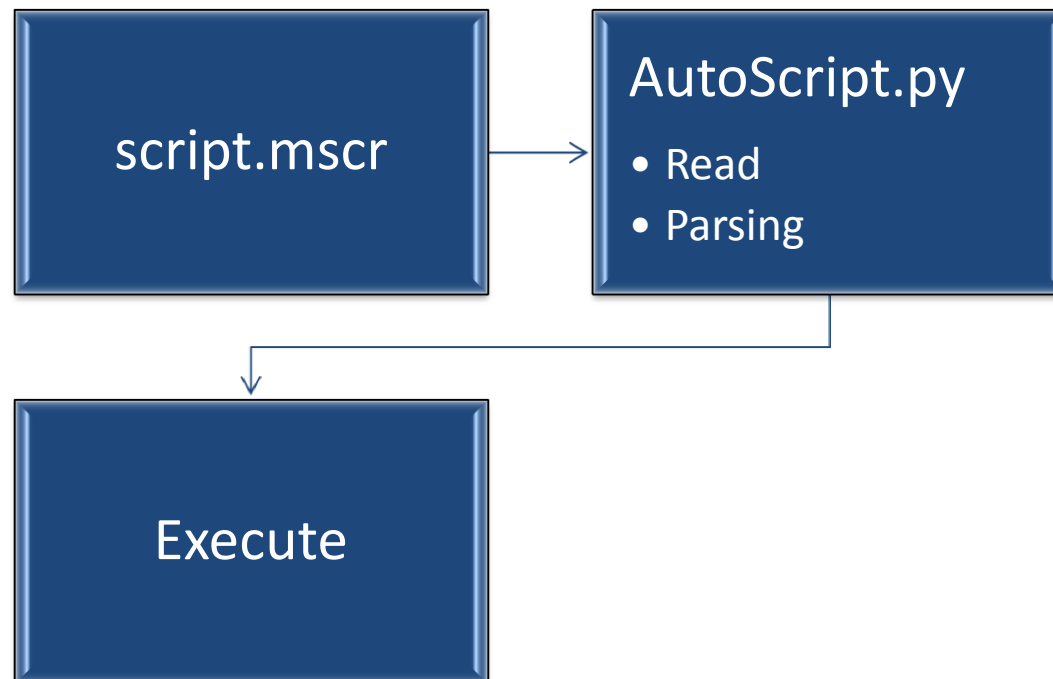
Bunun üçün OpenSource script ilə birlikdə özünüz haqqında informasiyanı [malware\\_lab@cert.gov.az](mailto:malware_lab@cert.gov.az)

ünvanına göndərməyiniz kifayətdir.

Script lazımı testlərdən keçdikdən sonra və əgər əlavə olunması məsləhət görülərsə sistemə əlavə ediləcək.

#### AutoScript.

- Bu modul sayəsində istifadəçilər modula xas şəkildə öz kiçik scriptlərini yazı biləcəklər.
- Modul hələlik əsas məqsəd olaraq cleaning əməliyyatlarını həyata keçirir.
- Lakin gələcəkdə təkmilləşdirilməsi nəzərdə tutulub.



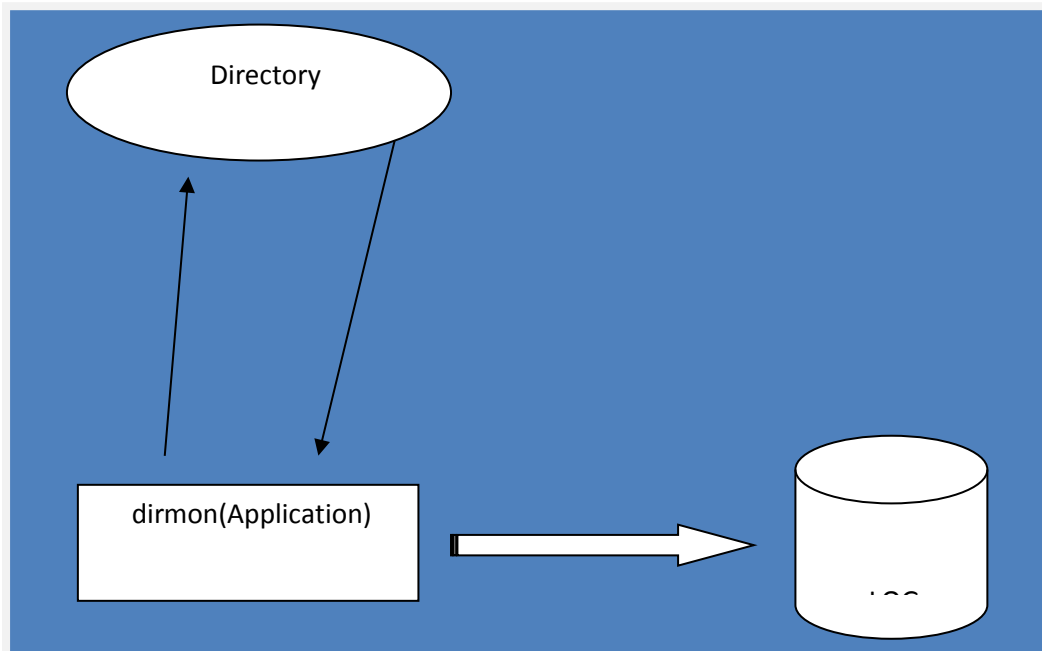
### Qovluq monitoring.

Qovluq monitoring (dirmon) özünə parametr olaraq verilmiş qovluqda baş verən dəyişiklikləri qeydə almaq və onları loglaşdırmaq üçün nəzərdə tutulmuşdur.

Hansı dəyişikliklər.

- Yeni yaradılan fayl
- Silinən fayl
- Adı dəyişdirilən fayl
- Modifikasiya edilmiş fayl

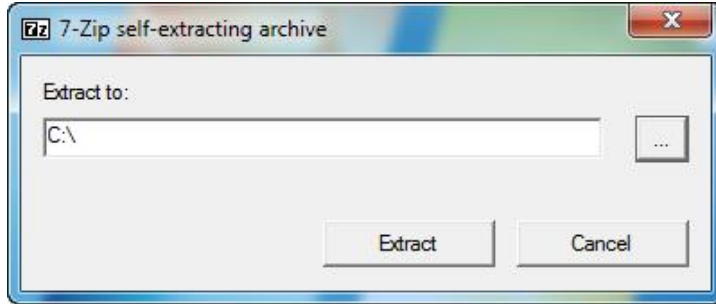
C proqramlaşdırma dilində yazılmışdır.



## MAK-ın yüklənməsi (Installation)

MAK 7Zip SFX ilə sıxışdırılmışdır.

Arxivi istədiyiniz qovluğa extract edə bilərsiniz.



Extract olunduqdan sonra qovluğun ilkin görünüşü.

Name	Date modified	Type	Size
common	12/31/2014 5:51 PM	File folder	
etc	12/31/2014 5:51 PM	File folder	
lib	1/4/2015 11:22 PM	File folder	
log	1/5/2015 12:56 AM	File folder	
plugins	12/31/2014 5:51 PM	File folder	
wd_process	12/31/2014 5:51 PM	File folder	
wd_registry	12/31/2014 5:51 PM	File folder	
autoscript	12/27/2014 3:52 PM	Python File	6 KB
autoscript	12/27/2014 4:21 PM	Registration Entries	1 KB
cc	12/27/2014 4:14 PM	Python File	3 KB
dialog.dll	11/18/2014 12:09 ...	Application extens...	14 KB
dirmon	12/22/2014 7:50 PM	Application	19 KB
main	12/30/2014 12:53 ...	Python File	25 KB
prototype	12/27/2014 10:43 ...	Python File	3 KB
webserv	1/5/2015 12:55 AM	Python File (no co...	2 KB
winstruct	12/27/2014 10:41 ...	Python File	11 KB

İlk olaraq qovlugin terribində olan autoscript.reg faylını editləmək tələb olunur.

Windows Registry Editor Version 5.00

[HKEY\_CLASSES\_ROOT\.mms]

[HKEY\_CLASSES\_ROOT\.mms\DefaultIcon]

@="C:\\MMS\\etc\\main.ico"

[HKEY\_CLASSES\_ROOT\.mms\shell]

[HKEY\_CLASSES\_ROOT\.mms\shell\open]

[HKEY\_CLASSES\_ROOT\.mms\shell\open\command]

@="\"python.exe\" \"C:\\MAK\\autoscript.py\" \"%1\""

Qirmizi rənglə qeyd olunmuş hissələrə MAK-ı install etdiyiniz qovluğun ünvanını (path) yazın.

Məsələn MAK sizde C:\Windows\MAK qovluğundadırsa həmin qovluğun tam adını daxil edin və slashlərə(‘\’) diqqət yetirin.

Bundan sonra dəyişiklikləri yadda saxlayıb həmin faylın üzərinə 2 dəfə klikləyərək Yes düyməsini vurun.

Artıq hazırdır.

Daha sonra etc qovluğunda ki, webserv.conf faylını açın və remoteden logları izləmək üçün listen IP ünvanı və port nömrəsini daxil edin.

Məsələn localda

127.0.0.1:8080

Bunu etdikdən sonra əsas qovluqda ki,webserv.pyw faylını işə salın daha sonra Brauzerinizdən

127.0.0.1:8080 adresinə qoşulun əgər aşağıdakı kimi bir səhifə ilə rastlaşsanız hər şey qaydasındadır Exit vuraraq çıxın.



Malware Müşahidə sistemi v1.0.0

[cert.gov.az](http://cert.gov.az)

[Exit](#)

Directory Monitor(Smart)

New Process Notification

Registry Change Notification(Smart)

Daha sonra əsas qovluqda olan cc.py faylını açın və Virustotal bölməsindəki vt\_apikey dəyərini daxil etməlisiniz. Virustotal Api key virustotal plugindən istifadə etmək üçündür. Bunun üçün ilk öncə virustotal saytında qeydiyyatdan keçin və sizə veriləcək apikeyi bura daxil edin.

## My API Key

### Basic information

This is your personal key, do not disclose it to anyone that you do not trust, do not embed it in scripts or software from which it can be easily retr about its confidentiality.

9f27495dcca8a2cae9edc

cc.py

```
#Virustotal
```

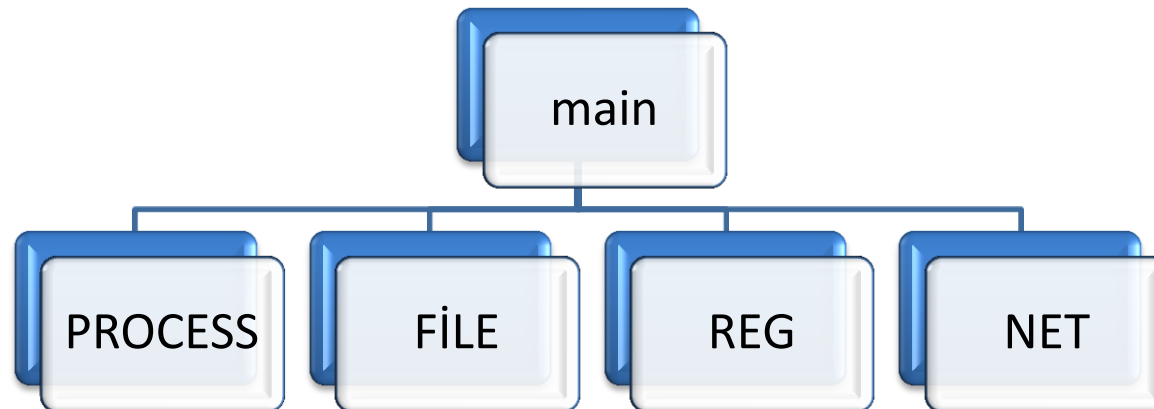
```
vt_apikey = '9f27495dcca8a2cae9edc
```

Dəyişiklikləri yadda saxlayın.

Artıq lazımı sazlamaları həyata keçirdiniz artıq sistem tam hazırdır.

[İstifadəçi təlimatları](#)

MAK-ın əsas hissəsini main modulu təşkil edir və digər modullar məhz bu modul altında birləşib. Main modulu 4 əsas alt sinifdən(classdan) ibarətdir.



İlkin olaraq konsoldan MAK-ın olduğu qovluğa keçin və pythonu işə salın daha sonra işə

```
from main import *
```

komandası ilə əsas modulu yükləyin.

Heç bir mesajla qarşılaşmadınızsa hər şey normaldır.

Əvvəl main modulunun özünün tərkibində olan əsas funksiyalardan danışaq.

 ps

Bu funksiya sistemdə cari vaxtda fəaliyyət göstərən proseslərin siyahısını bizə verir. Funksiya generator object yaradır. Object = (ProcessID ,ProcessName,ParentProcessID)

```
>>> from main import *
>>> for proc in ps():
...     proc
...
(4, 'System', 0)
(240, 'smss.exe', 4)
(388, 'csrss.exe', 336)
(440, 'wininit.exe', 336)
(448, 'csrss.exe', 432)
(496, 'services.exe', 440)
(528, 'winlogon.exe', 432)
(556, 'lsass.exe', 440)
(564, 'lsm.exe', 440)
(664, 'svchost.exe', 496)
(740, 'svchost.exe', 496)
(848, 'svchost.exe', 496)
(884, 'svchost.exe', 496)
(908, 'svchost.exe', 496)
(1068, 'svchost.exe', 496)
(1144, 'RtkAudioService.exe', 496)
(1180, 'RtHDVBg.exe', 1144)
(1200, 'svchost.exe', 496)
(1316, 'spoolsv.exe', 496)
(1352, 'svchost.exe', 496)
(1952, 'svchost.exe', 496)
(1936, 'dwm.exe', 884)
(1620, 'explorer.exe', 1396)
(396, 'taskhost.exe', 496)
(1064, 'SearchIndexer.exe', 496)
(2072, 'wmpnetwk.exe', 496)
(2256, 'svchost.exe', 496)
(3124, 'LMS.exe', 496)
(3188, 'svchost.exe', 496)
(2612, 'audiodg.exe', 848)
```

Artıq sistemdə fəaliyyət göstərən funksiyalar haqqında məlumat aldıq.

Artıq PROCESS sinifinə keçə bilərik.

# **1 PROCESS SINFI**

Sınıfə keçərkən initialize zamanı Prosesin ID nömrəsi tələb olunur.

```
>>> p = PROCESS(3136)
```

```
>>> p
```

```
<main.PROCESS object at 0x005848D0>
```

```
>>> dir(p)
```

```
['PEB', 'TIB', 'TSA', 'blackimport', 'dep', 'digest', 'dumppmemory', 'enum_handles', 'enum_threads', 'filetime', 'global_id', 'handle_name', 'karantin', 'module_info', 'modules', 'ntinfo', 'path', 'process_time', 'readmem', 'scan', 'signscan', 'systime', 'terminate', 'threadsuspend', 'threadterminate']
```

**path metodunun** köməkliyi ilə prosesin işə salındığı qovluğu görə bilərik. Bundan əlavə olaraq path metodu olduqca vacibdir digər metodların işləməsi üçün sinifi təyin etdikdən sonra mütləq ilk olaraq path metodunu işə salın.

```
>>> p.path()
```

```
'C:\\Program Files\\FastStone Capture\\FSCapture.exe'
```

Növbəti metod digest metodudur. Bu metod bizə faylın md5(default) və ya sha1 summasını geri qaytarır.

```
>>> p.digest()
```

```
'67DB9004AFB6AD0DC907D29EF5372F98'
```

```
>>> p.digest("sha1")
```

```
'0C7AC83169F7A2489D12498A5BABDF072031AF8E'
```

## **blackimport** metodu .

Bu metod faylın tərkibində etc\blacklist.txt faylının içərisində yerləşən və əksər hallarda Malware tərəfindən istifadə edilən funksiya adlarının bəziləri daxil edilib.

Əgər həmin funksiya adları faylın tərkibində mövcuddursa sistem bu haqda sizə məlumat verəcək.

```
>>> p.blackimport()
```

Yox əgər yoxdursa None geri qaytaracaq.

Digər metodumuz [modules](#) metodudur.

Bu metod proqram tərəfindən yüklənən modulları bizə ModuleEntry32 strukturunu generator object olaraq verir.

- ('dwSize',DWORD),
- ('th32ModuleID',DWORD),
- ('th32ProcessID',DWORD),
- ('GblcntUsage',DWORD),
- ('ProccntUsage',DWORD),
- ('modBaseAddr',BYTE),
- ('modBaseSize',DWORD),
- ('hModule',HMODULE),
- ('szModule',c\_char \* (MAX\_MODULE\_NAME32 + 1)),
- ('szExePath',c\_char \* MAX\_PATH)

```
>>> for mod in p.modules():
```

```
...     mod.szModule.decode(),mod.szExePath.decode()
```

```
...
```

```
('ntdll.dll', 'C:\\Windows\\SYSTEM32\\ntdll.dll')
```

```
('kernel32.dll', 'C:\\Windows\\system32\\kernel32.dll')
```

```
('KERNELBASE.dll', 'C:\\Windows\\system32\\KERNELBASE.dll')
```

```
('advapi32.dll', 'C:\\Windows\\system32\\advapi32.dll')
```

```
('msvcrt.dll', 'C:\\Windows\\system32\\msvcrt.dll')
```

```
('sechost.dll', 'C:\\Windows\\SYSTEM32\\sechost.dll')
```

```
('RPCRT4.dll', 'C:\\Windows\\system32\\RPCRT4.dll')
```

```
('avifil32.dll', 'C:\\Windows\\system32\\avifil32.dll')
```

```
('USER32.dll', 'C:\\Windows\\system32\\USER32.dll')
```

.....

### **dep metodu**

Bu metod cari proses üçün DEP-in aktiv olub olmadığını yoxlayır.

```
>>> p.dep()
DEP Disabled
```

process\_time metodu.

Bu metod prosesin işə düşmə vaxtını göstərir daha dəqiq FILE\_TIME strukturunu geri qaytarır.

```
>>> p.process_time()
```

```
[<winstruct.FILETIME object at 0x018A7C60>]
```

Oxumaq üçün isə **systemtime** metodundan istifadə etməlisiniz.

Bunun üçün FILETIME strukturunu bir dəyişəndə saxlayıb və ya bir başa systemtime metoduna ötürə bilərsiniz.

```
>>> p.systemtime(p.process_time())
```

```
['2015/1/5', '4:20:8']
```

**karantin metodu** bu metod isə normal bildiyiniz karantinlərdən biraz fərqli işləyir.

Malware tədqiqatçıları həmişə təhlükəli faylları sonradan analiz etmək üçün rezerv kopyasını götürürlər.

Bu metodda məhz bunun üçün nəzərdə tutulub.

Bu metodda 7Zip arxivləşdirmə proqramından istifadə olunub.

Cari faylı sıxışdırır + 'infected' parolu qoruması təyin edir və əsas qovluqda etc\qua qovluğunda faylın md5 summası adında saxlayır.

Bundan əlavə olaraq əməliyyat yerinə yetirildikdən sonra sizdən əsas faylı silmək istəyib istəmədiyinizi soruşacaq əgər Y düyməsini vursanız prosesi sonlandıracaq və faylı siləcək.

```
>>> p.karantin()
```

7-Zip 9.20 Copyright (c) 1999-2010 Igor Pavlov 2010-11-18

Scanning

Creating archive etc\qua\67DB9004AFB6AD0DC907D29EF5372F98.7z

Compressing FSCapture.exe

Everything is Ok

Delete file? [y or n] n

**enum\_threads metodu** generator obyekt olaraq ThreadEntry32 strukturunu qaytarır.

- ('dwSize',DWORD),
- ('cntUsage',DWORD),
- ('th32ThreadID',DWORD),
- ('th32OwnerProcessID',DWORD),
- ('tpBasePri',c\_long),

- ('tpDeltaPri',DWORD),
- ('dwFlags',DWORD)

```
>>> for th in p.enum_threads():
```

```
...     th.th32ThreadID
```

```
...
```

```
5832
```

```
4920
```

```
4516
```

```
5620
```

```
4440
```

**threadsuspend** **metodu** threadları müvəqqəti olaraq dayandırmaq və yenidən aktiv üçün nəzərdə tutulmuşdur.

Metod 2 parametr qəbul edir Bunlardan birincisi ThreadID digəri isə thread suspend yoxsa resume olacağıdır. Defoult olaraq resume = False. Əgər hər hansı problem meydana gəlsə return dəyəri -1.

```
>>> for th in p.enum_threads():
```

```
...     p.threadsuspend(th.th32ThreadID)
```

```
...
```

```
>>> for th in p.enum_threads():
```

```
...     p.threadsuspend(th.th32ThreadID,1)
```

```
...
```



**threadterminate metodu** threadları sonlandırmaq üçün nəzərdə tutulub.

Problem yaranarsa return dəyəri -1.

**PBI** metodu Prosesə aid əsas məlumatları PROCESS\_INFORMATION\_BLOCK Strukturunu geri qaytarır.

- ('Reserved1',c\_long),
- ('PebBaseAddress',c\_void\_p),#Convert to hex
- ('AffinityMask',c\_size\_t),
- ('BasePriority',c\_int),
- ('UniqueProcessId',c\_size\_t),
- ('InheritedFromUniqueProcessId',c\_size\_t)

```
>>> p.PBI().PebBaseAddress
```

```
2147315712
```

```
>>> hex(p.PBI().PebBaseAddress)
```

```
'0x7ffd7000'
```

**PEB metodu** Process Environment Blokunu oxumaq üçün nəzərdə tutulub.

Geriyə PROCESS\_ENVIRONMENT\_BLOCK strukturunu qaytarır.

- ("InheritedAddressSpace", BOOLEAN),
- ("ReadImageFileExecOptions", UCHAR),
- ("BeingDebugged", BOOLEAN),
- ("BitField", UCHAR),
- ("Mutant", HANDLE),
- ("ImageBaseAddress", c\_void\_p),
- ("Ldr", c\_void\_p),

- ("ProcessParameters", c\_void\_p),
- ("SubSystemData", c\_void\_p),
- ("ProcessHeap", c\_void\_p),
- ("FastPebLock", c\_void\_p),
- ("AtlThunkSListPtr", c\_void\_p),
- ("IFEOKey", c\_void\_p),
- ("CrossProcessFlags", DWORD),
- ("KernelCallbackTable", c\_void\_p),
- ("SystemReserved", DWORD),
- ("AtlThunkSListPtr32", c\_void\_p),
- ("ApiSetMap", c\_void\_p),
- ("TlsExpansionCounter", DWORD),
- ("TlsBitmap", c\_void\_p),
- ("TlsBitmapBits", DWORD \* 2),
- ("ReadOnlySharedMemoryBase", c\_void\_p),
- ("HotpatchInformation", c\_void\_p),
- ("ReadOnlyStaticServerData", c\_void\_p),
- ("AnsiCodePageData", c\_void\_p),
- ("OemCodePageData", c\_void\_p),
- ("UnicodeCaseTableData", c\_void\_p),
- ("NumberOfProcessors", DWORD),
- ("NtGlobalFlag", DWORD),
- ("CriticalSectionTimeout", LONGLONG),
- ("HeapSegmentReserve", DWORD),
- ("HeapSegmentCommit", DWORD),
- ("HeapDeCommitTotalFreeThreshold", DWORD),
- ("HeapDeCommitFreeBlockThreshold", DWORD),
- ("NumberOfHeaps", DWORD),
- ("MaximumNumberOfHeaps", DWORD),
- ("ProcessHeaps", c\_void\_p),
- ("GdiSharedHandleTable", c\_void\_p),
- ("ProcessStarterHelper", c\_void\_p),
- ("GdiDCAttributeList", DWORD),
- ("LoaderLock", c\_void\_p),
- ("OSMajorVersion", DWORD),

- ("OSMinorVersion", DWORD),
- ("OSBuildNumber", WORD),
- ("OSCSVersion", WORD),
- ("OSPlatformId", DWORD),
- ("ImageSubsystem", DWORD),
- ("ImageSubsystemMajorVersion", DWORD),
- ("ImageSubsystemMinorVersion", DWORD),
- ("ActiveProcessAffinityMask", DWORD),
- ("GdiHandleBuffer", DWORD \* 34),
- ("PostProcessInitRoutine", PPS\_POST\_PROCESS\_INIT\_ROUTINE),
- ("TlsExpansionBitmap", c\_void\_p),
- ("TlsExpansionBitmapBits", DWORD \* 32),
- ("SessionId", DWORD),
- ("AppCompatFlags", ULONGLONG),
- ("AppCompatFlagsUser", ULONGLONG),
- ("pShimData", c\_void\_p),
- ("AppCompatInfo", c\_void\_p),
- ("CSDVersion", UNICODE\_STRING),
- ("ActivationContextData", c\_void\_p),
- ("ProcessAssemblyStorageMap", c\_void\_p),
- ("SystemDefaultActivationContextData", c\_void\_p),
- ("SystemAssemblyStorageMap", c\_void\_p),
- ("MinimumStackCommit", DWORD),
- ("FlsCallback", c\_void\_p),
- ("FlsListHead", LIST\_ENTRY),
- ("FlsBitmap", c\_void\_p),
- ("FlsBitmapBits", DWORD \* 4),
- ("FlsHighIndex", DWORD),
- ("WerRegistrationData", c\_void\_p),
- ("WerShipAssertPtr", c\_void\_p),
- ("pContextData", c\_void\_p),
- ("pImageHeaderHash", c\_void\_p),
- ("TracingFlags", DWORD)

```
>>> peb_address = hex(p.PBI().PebBaseAddress)
>>> p.PEB(peb_address)
<winstruct.PROCESS_ENVIRONMENT_BLOCK object at 0x01A97F80>
>>> p.PEB(peb_address).BeingDebugged
0
```

### **fileinfo metodu.**

Bu metod fayl ölçüsü və filetime haqqında məlumatları verir.

```
>>> p.fileinfo()
#####
File size:2488320
File create time:2014-12-27 13:49:54
File modified time:2014-12-27 13:49:54
Access time:2014-12-31 17:06:14.412818
#####
```

## signscan metodu

Bu metod etc\db\_signature.txt faylında olan baytları faylın tərkibində axtarır və əgər nəticə mövcuddursa signaturaya uyğun gələn makro adı geri qaytarır .

Məsələn.

The screenshot shows a Notepad++ window with a hex-to-ASCII conversion table. The table has columns for Offset, Hex, and Ascii. The hex data is as follows:

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	Ascii
00172F60	46	00	69	00	6E	00	64	00	3A	00	20	00	43	00	61	00	F.i.n.d...C.a.
00172F70	6E	00	27	00	74	00	20	00	66	00	69	00	6E	00	64	00	n.'t...f.i.n.d.
00172F80	20	00	74	00	68	00	65	00	20	00	74	00	65	00	78	00	...t.h.e...t.e.x.
00172F90	74	00	20	00	22	00	00	00	22	00	00	00	00	00	00	00	t..."
00172FA0	46	00	69	00	6E	00	64	00	3A	00	20	00	49	00	6E	00	F.i.n.d...I.n.
00172FB0	76	00	61	00	6C	00	69	00	64	00	20	00	72	00	65	00	v.a.l.i.d...r.e.
00172FC0	67	00	75	00	6C	00	61	00	72	00	20	00	65	00	78	00	g.u.l.a.r...e.x.
00172FD0	70	00	72	00	65	00	73	00	73	00	69	00	6F	00	6E	00	p.r.e.s.s.i.o.n.
00172FE0	00	00	00	00	5E	20	7A	65	72	6F	20	6C	65	6E	67	74	...^...zero lengt
00172FF0	68	20	6D	61	74	63	68	00	52	00	65	00	78	00	6C	00	h.match.Rep.l.
00173000	61	00	63	00	65	00	3A	00	20	00	52	00	65	00	70	00	a.c.c...Rep.
00173010	6C	00	61	00	63	00	65	00	64	00	20	00	74	00	68	00	l.a.c.e.d...t.h.
00173020	65	00	20	00	31	00	73	00	74	00	20	00	6F	00	63	00	e...l.s.t...o.c.
00173030	63	00	75	00	72	00	72	00	65	00	6E	00	63	00	65	00	c.u.r.r.e.n.c.e.
00173040	20	00	66	00	72	00	6F	00	6D	00	20	00	74	00	68	00	...f.r.o.m...t.h.
00173050	65	00	20	00	74	00	6F	00	70	00	2E	00	20	00	54	00	e...t.o.p...T.
00173060	68	00	65	00	20	00	65	00	6E	00	64	00	20	00	6F	00	h.e...e.n.d...o.
00173070	66	00	20	00	64	00	6F	00	63	00	75	00	6D	00	65	00	f...d.o.c.u.m.e.
00173080	6E	00	74	00	20	00	68	00	61	00	73	00	20	00	62	00	n.t...h.a.s...b.
00173090	65	00	65	00	6E	00	20	00	72	00	65	00	61	00	63	00	e.e.n...r.e.a.c.
001730A0	68	00	65	00	64	00	2E	00	00	00	00	00	00	00	00	00	h.e.d.....

A red arrow points from the hex data to the command prompt, which shows the following command:

```
>>> p.signscan()  
Notepad++  
>>>
```

Db\_signature faylına özünüz istədiyiniz imzaları əlavə edə bilərsiniz.

Name:signature şəklində.

## TIB metodu.

Bu metod Threadlar haqqında informasiya daha dəqiq Thread\_Information\_Block strukturunu geri qaytarır.

Parametr olaraq ThreadID dəyərini qəbul etməlidir.

- ('ExitStatus', NTSTATUS),
- ('TebBaseAddress', c\_void\_p),
- ('ClientId', CLIENT\_ID),
- ('AffinityMask', c\_size\_t),
- ('Priority', c\_int32),
- ('BasePriority', c\_int32)

```
>>> for k in p.enum_threads():
```

```
...     k.th32ThreadID
```

```
5732
```

```
3304
```

```
>>> p.TIB(5732)
```

```
<winstruct.TIB object at 0x019ACF80>
```

```
>>> p.TIB(5732).TebBaseAddress
```

```
2147348480
```

## TSA metodu.

Bu metod parametr olaraq qəbul etdiyi Thread(ID)-in başlanğıc adresini geri qaytarır.

```
>>> hex(p.TSA(5732))  
'0x136ddaf'
```

Thread ID	Start Address	Start Name
5732	1,053,153	notepad++.exe+0x136ddaf

**enum\_handles metodu** bu metod Prosesin əldə etdiyi handları extract etmək üçün nəzərdə tutulub.

Geri SYSTEM\_HANDLE\_INFORMATION strukturunu generator object qaytarır.

- ('ProcessID',c\_ushort),
- ('CreateBackTrackIndex',c\_ushort),
- ('ObjectTypeNumber',c\_ubyte),
- ('Flags',c\_ubyte),
- ('Handle',c\_ushort),
- ('ObjectAddr',c\_void\_p),
- ('AccessMask',c\_int)

```
>>> for handles in p.enum_handles():
```

```
...     hex(handles.Handle)
```

```
...
```

```
'0x4'
```

```
'0x8'
```

```
'0xc'
```

```
'0x10'
```

```
'0x14'
```

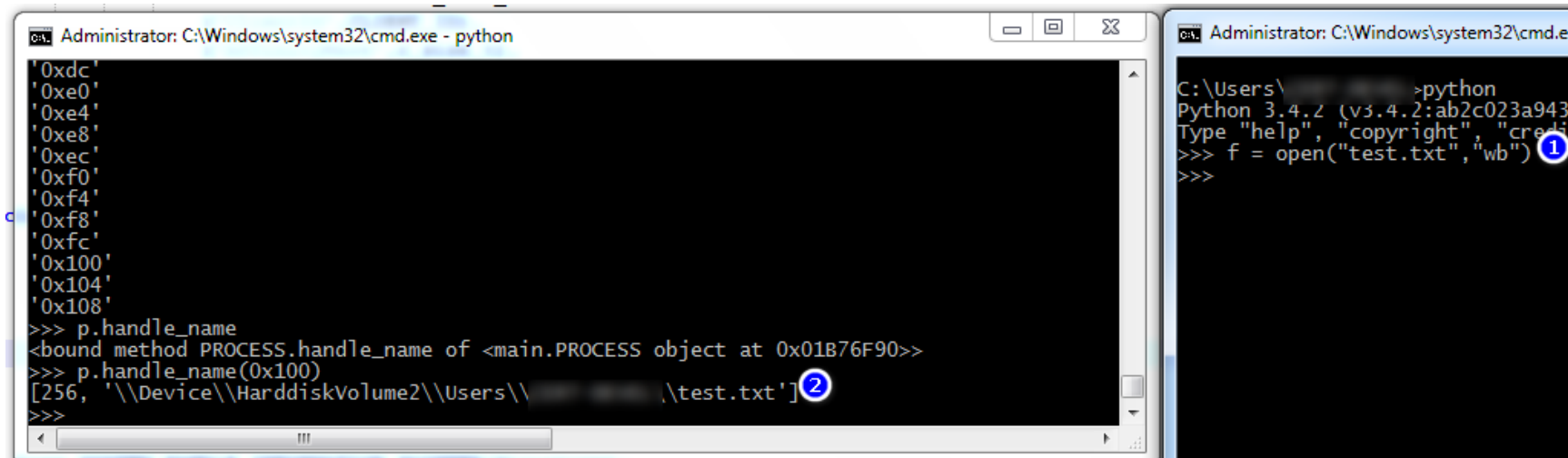
```
'0x18'
```

### handle\_name metodu.

Bu metod proses tərəfindən açılmış handle haqqında əsas məlumatı geri qaytarır.

Məsələn PROCESS sinifinə daxil edilmiş Digər python interpretoru tərəfindən yaradılmış faylı MAK ilə handle adını extract edirik.

Parametr olaraq enum\_handles.Handle dəyərini hex olaraq qəbul etməlidir.



```
C:\> Administrator: C:\Windows\system32\cmd.exe - python
'0xdc'
'0xe0'
'0xe4'
'0xe8'
'0xec'
'0xf0'
'0xf4'
'0xf8'
'0xfc'
'0x100'
'0x104'
'0x108'
>>> p.handle_name
<bound method PROCESS.handle_name of <main.PROCESS object at 0x01B76F90>>
>>> p.handle_name(0x100)
[256, '\\Device\\HarddiskVolume2\\Users\\[redacted]\\test.txt']
>>>

C:\> Administrator: C:\Windows\system32\cmd.exe
C:\Users\[redacted]> python
Python 3.4.2 (v3.4.2:ab2c023a943)
Type "help", "copyright", "credits()" or "quit()" for more
>>> f = open("test.txt", "wb")
>>>
```

### dumpmemory metodu.

Bu metod isə adından da aydın olduğu kimi Prosesin məxsus yaddaşın dump edilməsi üçün nəzərdə tutulmuşdur.

Parametr olaraq 0(Default) və 1 dəyərlərini qəbul edir.

0 = MiniDumpNormal(Normal Crash zamanı məhz bu tiptən istifadə edilir).

1 = MiniDumpWithFullMemory

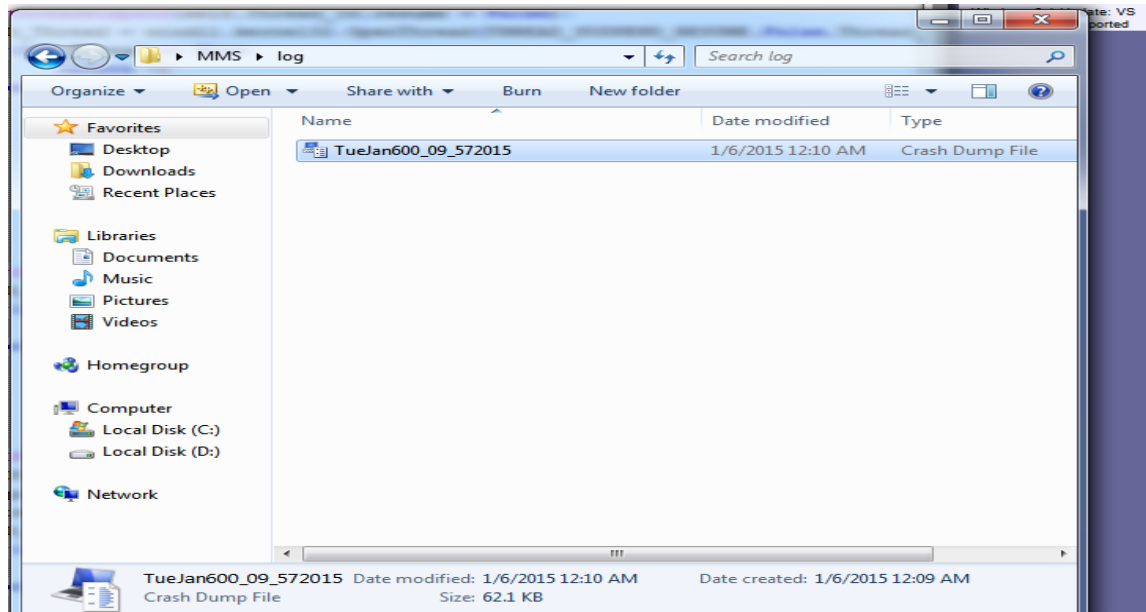


Ən çox istifadə edilən dump tipləri istifadə olunub.Ətraflı məlumat üçün

<http://msdn.microsoft.com/en-us/library/windows/desktop/ms680519%28v=vs.85%29.aspx>

```
>>> p.dumpmemory()  
Successfully dumped  
1  
>>>
```

Dump faylı log qovluğunda yaradılır.



Try it! It's just one more way that OSR helps the Windows dri

#### How To Upload Your Crash Dump for Instant .

Select a dump file to upload using the button below. Your upl

- A mini-dump file
- A kernel summary dump
- A ZIP archive including a mini-dump or kernel summary in the analysis.

*We strongly urge you to zip your dumps for uploading.* The up accept uploads up to about 40MB in size. Files larger than th simply not upload successfully.

Click "Upload Dump." Your file will be uploaded to our server v minute or so) the analysis output will be displayed in your bro

Dump File:  TueJan600\_09\_572015.dmp

NOTE: By clicking the "Upload Dump" button, you agree that including as an example in OSR's Windows System Software property of OSR Open Systems Resources, Inc. Your dump t that associates your dump with your IP address or any other i

File Edit View History Bookmarks Tools Help

Thanks for your submission

www.osronline.com/dump/DA2.cfm

76de0000 76fdb000 wininet wininet.dll  
76de0000 76fdb000 iertutil iertutil.dll  
76fe0000 77080000 advapi32 advapi32.dll  
77080000 771bc000 ntdll ntdll.dll  
771f0000 771f5000 psapi psapi.dll  
77200000 77235000 ws2\_32 ws2\_32.dll  
77240000 77297000 shlwapi shlwapi.dll  
772a0000 772a6000 nsi nsi.dll

**Raw Stack Contents**

Memory access error at 'StackLimit' @(((ntdll!\_NT\_TIB \*)(\$teb)->StackBase)'

**Dump Header Information**

----- User Mini Dump Analysis

MINIDUMP\_HEADER:  
Version A793 (61B1)  
NumberOfStreams 7  
Flags 0  
0000 MiniDumpNormal

Streams:  
Stream 0: type ThreadListStream (3), size 00000394, RVA 00000194  
19 threads  
RVA 00000198, ID EE0, Teb:000000007FFDF000  
RVA 000001C8, ID B7C, Teb:000000007FFDE000  
RVA 000001F8, ID EA8, Teb:000000007FFDD000  
RVA 00000228, ID FAC, Teb:000000007FFDC000  
RVA 00000258, ID D10, Teb:000000007FFD4000  
RVA 00000288, ID AF0, Teb:000000007FFAE000  
RVA 000002B8, ID 14CC, Teb:000000007FFDB000  
RVA 000002E8, ID 14D0, Teb:000000007FFDA000  
RVA 00000318, ID 14D4, Teb:000000007FFD9000  
RVA 00000348, ID 14D8, Teb:000000007FFD8000  
RVA 00000378, ID 1984, Teb:000000007FFD7000  
RVA 000003A8, ID 1988, Teb:000000007FFD6000  
RVA 000003D8, ID 19B8, Teb:000000007FFA9000  
RVA 00000408, ID 19BC, Teb:000000007FFA8000  
RVA 00000438, ID 1DE4, Teb:000000007FFAB000  
RVA 00000468, ID 1EF8, Teb:000000007FFAA000  
RVA 00000498, ID 1AA4, Teb:000000007FFAD000  
RVA 000004C8, ID 1668, Teb:000000007FFD3000  
RVA 000004F8, ID 13E0, Teb:000000007FFAF000  
Stream 1: type ModuleListStream (4), size 000019C0, RVA 00000534  
61 modules  
RVA 00000538, 00050000 - 00221000: 'C:\Windows\System32\Macromed\Flash\FlashPlayerPlugin\_16\_0\_0\_235.exe'  
RVA 000005A4, 77080000 - 771bc000: 'C:\Windows\System32\ntdll.dll'  
RVA 00000610, 75880000 - 75954000: 'C:\Windows\System32\kernel32.dll'  
RVA 0000067C, 75250000 - 7529a000: 'C:\Windows\System32\KERNELBASE.dll'  
RVA 000006E8, 76ce0000 - 76dd5000: 'C:\Windows\System32\wininet.dll'  
RVA 00000754, 75740000 - 7587a000: 'C:\Windows\System32\user32.dll'

### **module\_info metodu.**

Bu metod proses tərəfindən yüklənən modullar haqqında informasiyanı(MODULE\_INFO Strukturunu) geri qaytarır.

- ('lpBaseAddress',c\_void\_p),
- ('SizeOfImage',c\_ulong),
- ('EntryPoint',c\_void\_p)

```
>>> for mod in p.modules():
```

```
...     mod.szModule,mod.szExePath,p.module_info(mod.hModule).SizeOfImage
```

```
...
```

```
(b'ntdll.dll', b'C:\\Windows\\SYSTEM32\\ntdll.dll', 1294336)
```

```
(b'kernel32.dll', b'C:\\Windows\\system32\\kernel32.dll', 868352)
```

```
(b'KERNELBASE.dll', b'C:\\Windows\\system32\\KERNELBASE.dll', 303104)
```

### **readmem metodu.**

Bu metod prosesin yaddasınнан məlumat oxumaq üçün nəzərdə tutulub.

Parametr olaraq oxunulacaq yaddaş adresini və məlumat uzunluğunu qəbul edir.

Geriyə məlumatı raw modda qaytarır.

Məlumat yoxdursa geriye 0 dəyərini qaytarır.

## 2 FILE SINFI

### \_hash metodu.

Bu metod özünə parametr olaraq gələn faylın(Path) md5 summasını geri qaytarır.

```
>>> f = FILE  
  
>>> f._hash("c:\\Windows\\explorer.exe")  
  
'40D777B7A95E00593EB1568C68514493'
```

### fuzzy metodu.

Bu metod zamanı ssdeep proqram təminatının apisindən istifadə olunub.

Ssdeep faylı hissələrə bölərək onlara uyğun hash dəyəri yaradır.

Ssdeep-in əsas məqsədi 2 eyni fayldan 1-i kiçik dəyişikliyə uğradığı zamanı bu fayllar arasında uyğunluq olub olmadığını yoxlayır.

Malware Foresics zamanı geniş istifadə edilir.

Ətraflı məlumat üçün.

<http://ssdeep.sourceforge.net/>

```
>>> file1 = f.fuzzy("c:\\Windows\\explorer.exe")
```

```
>>> file1
```

```
b'49152:zPWgRz3y/Ny+DPmvYYYYYYYYYYYRYYYYYYYYYYYE3iA7/eFUJN9ojoso2xnon:T/3yFylvYYYYYYYYYYYRYYYYYYY  
YYE5'
```

Daha sonra isə digər faylın hash summasını çıxardıb `fuzzy_compare` metodu ilə bunları müqayisə edirsiniz və sizə fayllar arasında uyğunluğu gösterecek.

```
>>> file2 = f.fuzzy("c:\\Windows\\explorer.exe")
```

```
>>> f.fuzzy_compare(file1,file2)
```

```
100
```

### **shred metodu.**

Bu metod lazım bildiyiniz faylları xüsusi metodla silməyə imkan verir.

Əsas məqsəd faylların Recover proqram təminatlarının köməkliyi ilə geri qaytarılmasının qarşısını almaqdır.

Metod faylın üzərinə 3 dəfə 2-si random 1 i \x0 olmaq şərti ilə məlumat yazır və faylı silir.

Metod parametr olaraq fayl path qəbul edir.

## pipes metodu.

Bu metod Sistemdə aktiv olan pipləri extract edir.

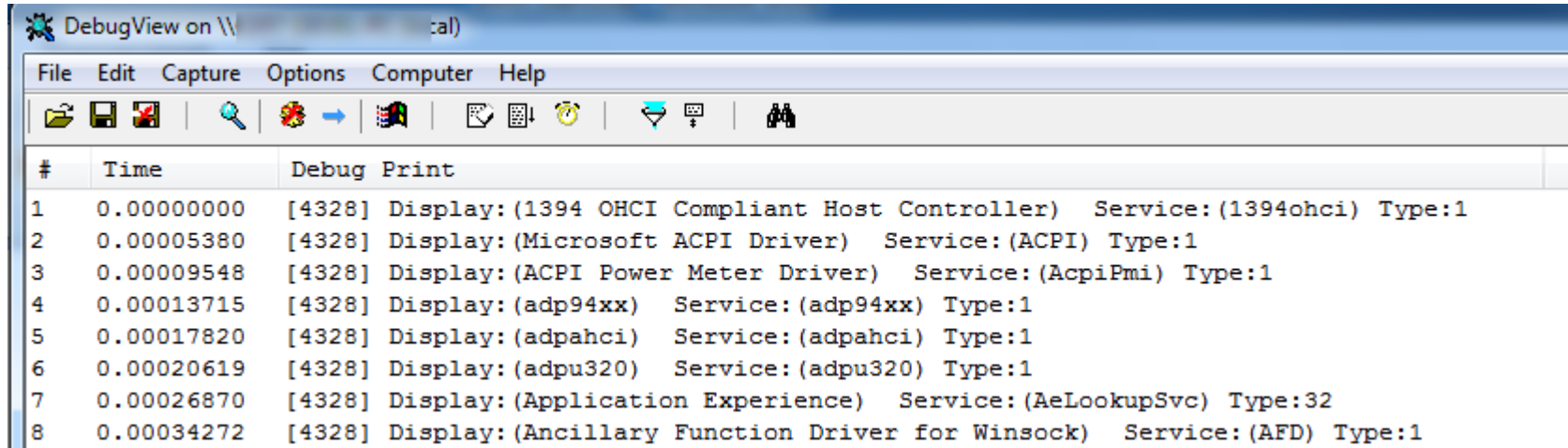
```
>>> f.pipes()
***** PIPE object *****
lsass
protected_storage
ntsvcs
scerpc
plugplay
winsock2\CatalogChangeListener-2e4-0
epmapper
winsock2\CatalogChangeListener-1b8-0
LSM_API_service
eventlog
winsock2\CatalogChangeListener-350-0
atsvc
winsock2\CatalogChangeListener-38c-0
wkssvc
keysvc
trkwks
srvsvc
winsock2\CatalogChangeListener-1f0-0
winsock2\CatalogChangeListener-22c-0
MsFteWds
browser
PIPE_EVENTROOT\CIMV2SCM EVENT PROVIDER
w32time_alt
ProtectedPrefix\LocalService\FTHPIPE
gecko-crash-server-pipe.4080
chrome.4080.f673860.1497288812
chrome.Flash8080.5EAE6188.8532
chrome.Flash8080.5EAE6188.8777
```

### enumservices metodu.

Bu metod sistemdeki Servisleri çıkarır.

Metod neticeleri output olarak Debug-a gönderir.

Neticeleri görmek için SysInternalsın DbgView utility-sinden istifade edin.

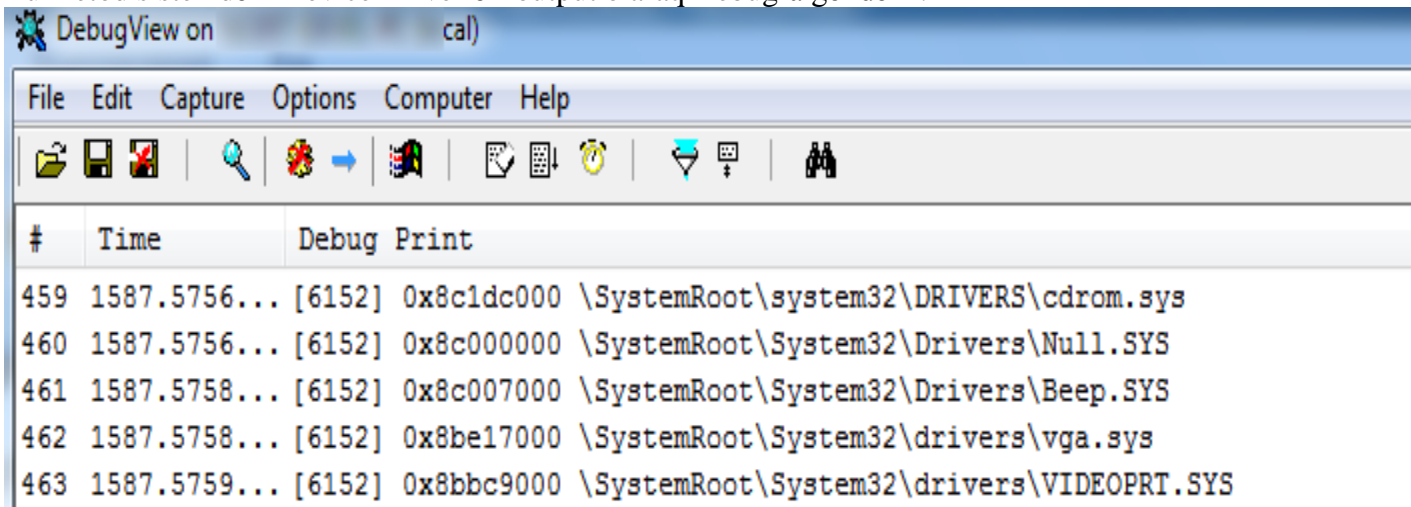


The screenshot shows the DebugView utility window with the following data:

#	Time	Debug Print
1	0.00000000	[4328] Display:(1394 OHCI Compliant Host Controller) Service:(1394ohci) Type:1
2	0.00005380	[4328] Display:(Microsoft ACPI Driver) Service:(ACPI) Type:1
3	0.00009548	[4328] Display:(ACPI Power Meter Driver) Service:(AcpiPmi) Type:1
4	0.00013715	[4328] Display:(adp94xx) Service:(adp94xx) Type:1
5	0.00017820	[4328] Display:(adpahci) Service:(adpahci) Type:1
6	0.00020619	[4328] Display:(adpu320) Service:(adpu320) Type:1
7	0.00026870	[4328] Display:(Application Experience) Service:(AeLookupSvc) Type:32
8	0.00034272	[4328] Display:(Ancillary Function Driver for Winsock) Service:(AFD) Type:1

### enumdd metodu.

Bu metod sistemdeki Device Driverleri output olarak Debug-a gönderir.



The screenshot shows the DebugView utility window with the following data:

#	Time	Debug Print
459	1587.5756...	[6152] 0x8c1dc000 \SystemRoot\system32\DRIVERS\cdrom.sys
460	1587.5756...	[6152] 0x8c000000 \SystemRoot\System32\Drivers\Null.SYS
461	1587.5758...	[6152] 0x8c007000 \SystemRoot\System32\Drivers\Beep.SYS
462	1587.5758...	[6152] 0x8be17000 \SystemRoot\System32\drivers\vga.sys
463	1587.5759...	[6152] 0x8bbc9000 \SystemRoot\System32\drivers\VIDEOPRT.SYS

## 3 NET SİNFİ

Bu sınıfta şəbəkə ilə bağlı funksiyalar yer alıb.

İlk **metodumuz TCP**.

Bu metod aktiv TCP qoşulmalarını göstərir.

```
>>> n = NET
>>> n.TCP()

=====
127.0.0.1      51023      127.0.0.1      51024      State:TCP_STATE_SYN_RCVD
127.0.0.1      51024      127.0.0.1      51023      State:TCP_STATE_SYN_RCVD
127.0.0.1      40002      127.0.0.1      40002      State:TCP_STATE_SYN_RCVD
127.0.0.1      12350      127.0.0.1      12350      State:TCP_STATE_SYN_RCVD
127.0.0.1      443       127.0.0.1      443       State:TCP_STATE_SYN_RCVD
127.0.0.1      443       127.0.0.1      443       State:TCP_STATE_SYN_RCVD
127.0.0.1      443       127.0.0.1      443       State:TCP_STATE_SYN_RCVD
127.0.0.1      443       127.0.0.1      443       State:TCP_STATE_SYN_RCVD
=====
```

### UDP Metodu.

Bu metod qoşulma gözləyən(Listening) UDP Socketləri göstərir.

```
>>> n.UDP()

=====
137
137
138
138
1900
1900
1900
1900
56065
56066
56954
56955
56956
65080
=====
```



### **adapterinfo** metodu.

Bu metod Əməliyyat sistemi tərəfindən istifadə olunan Network Adapterləri haqqında məlumatı generator obyekt olaraq qaytarır.

```
>>> for i in n.adapterinfo():
...     i
('192.168.56.1', '{0E6EC273-C6EF-4861-8B47-060844361B50}', '0.0.0.0', '', 'VirtualBox Host-Only Ethernet Adapter', 'Realtek PCIe GBE Family Controller')
```

Və son olaraq **hosts** metodu bu metod hosts faylının tərkibini ekrana yazdırır.

```
>>> n.hosts()

1      # Copyright (c) 1993-2009 Microsoft Corp.
2      #
3      # This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
4      #
5      # This file contains the mappings of IP addresses to host names. Each
6      # entry should be kept on an individual line. The IP address should
7      # be placed in the first column followed by the corresponding host name
8      # The IP address and the host name should be separated by at least one
9      # space.
10     #
11     # Additionally, comments (such as these) may be inserted on individual
12     # lines or following the machine name denoted by a '#' symbol.
13     #
14     # For example:
15     #
16     #      102.54.94.97      rhino.acme.com      # source server
17     #      38.25.63.10      x.acme.com         # x client host
18
19     # localhost name resolution is handled within DNS itself.
20     #      127.0.0.1      localhost
21     #      ::1           localhost
```

## 4 REG SİNFİ

Bu sınıfta 1 ədəd metod mövcuddur.

**dump** metodu.

Metod etc\regdump.txt faylının əsasında işliyin. Beləki faylın tərkibi aşağıda ki şəkildə göstərilmişdir.

```
hklm_run->HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
hklm_runonce->HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce
rename_files->HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RenameFiles->Keys
ifeo->HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options->Keys
```

Macro ad və Registry Key mövcuddur metod parametrlər olaraq makro ad qəbul edir.

Və həmin açarı dump edir.

```
>>> r.dump("hklm_run")
```

```
SOFTWARE\Microsoft\Windows\CurrentVersion\Run
```

```
#####
```

```
Subkey 0 : values 1 : keyLastModifiedTime: 130650254290211704
```

```
#####
```

```
test:test.exe(REG_SZ)
```

Bundan əlavə olaraq funksiya subkey adında 2ci parametrlə qəbul edir. Defolt olaraq 0 dəyəridir. 1 edildiyi təqdirdə makro adda verilən Registry keyə məxsus subkeyləri dump edir.

```
>>> r.dump("ifeo",1)
```

```
SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options
```

```
#####
```

```
Subkey 7 : values 0 : keyLastModifiedTime: 130645504908226308
```

```
#####
```

```
DllNXOptions
```

```
FlashPlayerApp.exe
```

```
FlashPlayerPlugin_16_0_0_235.exe
```

```
FlashPlayerUpdateService.exe
```

```
FlashUtil32_16_0_0_235_Plugin.exe
```

```
IEInstal.exe
```

```
taskmgr.exe
```

Faylın tərkibinə uyğun şəkildə dəyişiklik edə bilərsiniz.

Yəni öz makro adlarınızı və Registry keylərinizi daxil edə bilərsiniz.

## Müşahidə.

Sistem müşahidə aparmaq üçün 3 əsas modul təklif edir.

1. Registry Monitoring
2. Directory Activity Monitoring
3. New Process Activity

### 1.Registry Monitoring.

Bu modul wd\_registry qovluğunun tərkibində olan Keys.txt faylının əsasında işləyir.

Faylın tərkibi.

HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run

HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft

Özünü əlavə açarlar daxil edə bilərsiniz.

Modul bu açarları izləməyə alır.

Açarların sayı qədər Thread-dan istifadə edilir.Ona görə də vacib açarları daxil etməyiniz məsləhətdir.

CPU və yaddaş testlərindən keçib.Tam stabildir.

Beleliklə modul Keys.txt faylının içərisində olan açarlarda baş vermiş dəyişiklikləri([+]=Əlavə ediləni göstərir,[-] Silinəni göstərir) loglaşdırır.

Modul Multi-Threadingdən istifadə etdiyi üçün prosesi sonlandırmaq üçün MAK-ın içərisindən **KillThread** funksiyasını işə salmalısınız.

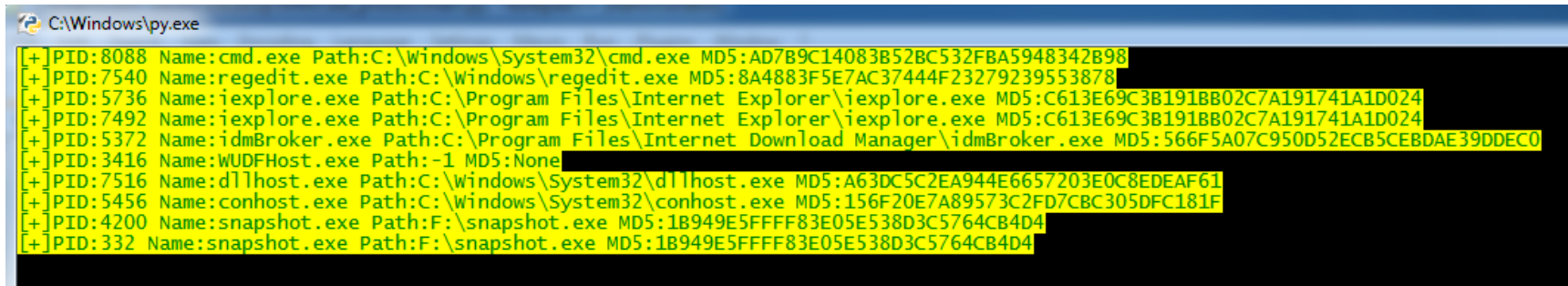
Beləki modul MAK adında Event yaradır və WaitForSingleObject istifadə edərək prosesi sonlandırmaq üçün signal gözləyir.

KillThread funksiyası ilə bu signalı verərək Modulun işini yekunlaşdırı bilərsiniz.

```
>>> KillThread(  
OK Thread stopped  
1
```

## 2.New Process Monitoring.

Bu modul Yeni işə düşən proseslər haqqında məlumat verir və loglaşdırır.



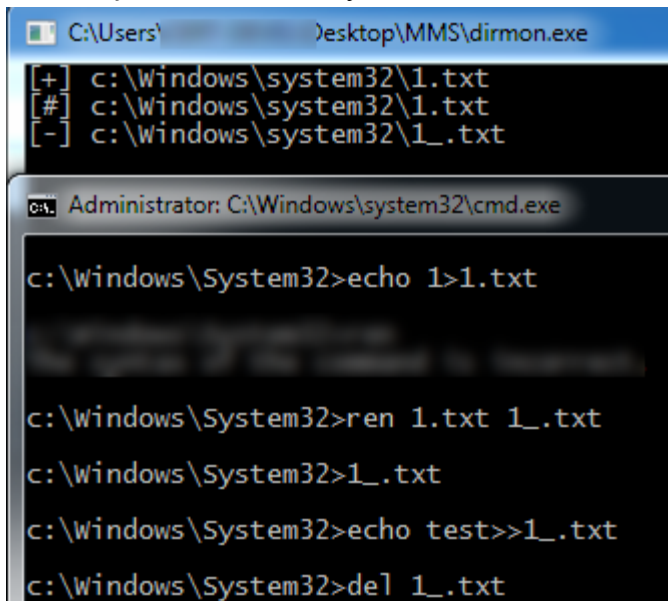
```
C:\Windows\py.exe
[+] PID:8088 Name:cmd.exe Path:C:\Windows\System32\cmd.exe MD5:AD7B9C14083B52BC532FBA5948342B98
[+] PID:7540 Name:regedit.exe Path:C:\Windows\regedit.exe MD5:8A4883F5E7AC37444F23279239553878
[+] PID:5736 Name:iexplore.exe Path:C:\Program Files\Internet Explorer\iexplore.exe MD5:C613E69C3B191BB02C7A191741A1D024
[+] PID:7492 Name:iexplore.exe Path:C:\Program Files\Internet Explorer\iexplore.exe MD5:C613E69C3B191BB02C7A191741A1D024
[+] PID:5372 Name:idmBroker.exe Path:C:\Program Files\Internet Download Manager\idmBroker.exe MD5:566F5A07C950D52ECB5CEBDAE39DDEC0
[+] PID:3416 Name:WUDFHost.exe Path:-1 MD5:None
[+] PID:7516 Name:dllhost.exe Path:C:\Windows\System32\dllhost.exe MD5:A63DC5C2EA944E6657203E0C8EDEAF61
[+] PID:5456 Name:conhost.exe Path:C:\Windows\System32\conhost.exe MD5:156F20E7A89573C2FD7CBC3050FC181F
[+] PID:4200 Name:snapshot.exe Path:F:\snapshot.exe MD5:1B949E5FFFF83E05E538D3C5764CB4D4
[+] PID:332 Name:snapshot.exe Path:F:\snapshot.exe MD5:1B949E5FFFF83E05E538D3C5764CB4D4
```

## 3.Directory Activity Monitoring

Bu modul C proqramlaşdırma dilində yazılıb.Əsas məqsəd özünə parametr olaraq gələn qovluqda baş verən dəyişiklikləri istifadəçiyə bildirməkdir və qeydə alınan dəyişiklikləri loqlaşdırmaqdır.

Modulu Birbaşa MAK –ın içərisindən işə sala bilərsiniz.

Bunun üçün dirmon funksiyasından istifadə edə bilərsiniz.



```
C:\Users\...desktop\MMS\dirmon.exe
[+] c:\Windows\system32\1.txt
[#] c:\Windows\system32\1.txt
[-] c:\Windows\system32\1_.txt

Administrator: C:\Windows\system32\cmd.exe
c:\Windows\System32>echo 1>1.txt

c:\Windows\System32>ren 1.txt 1_.txt
c:\Windows\System32>1_.txt

c:\Windows\System32>echo test>>1_.txt
c:\Windows\System32>del 1_.txt
```

## Pluginlər

Pluginlər MAK-a əlavə olaraq yüklənir.

Əlavə etmək üçün

```
from plugins.pluginadi import *
```

### enumfiles Plugini

```
from plugins.enumfiles import *
```

EnumFiles funksiyası.

Bu plugin parametrlər olaraq qəbul etdiyi qovluqda ki, faylları sıralayaraq ekrana yazır.

Normal üsullardan fərqi Native Api əsasında işləməsidir.

```
type help , copyright , credits or license for more information.
>>> from plugins.enumfiles import *
>>>
>>> EnumFiles("c:\\windows")
FileAttr:16 CreateTime:128920126259532869 FileName:..
FileAttr:16 CreateTime:128920126259532869 FileName:..
FileAttr:16 CreateTime:128920207509229246 FileName:addins
FileAttr:16 CreateTime:128920126259532869 FileName:AppCompat
FileAttr:16 CreateTime:128920126259532869 FileName:AppPatch
FileAttr:21 CreateTime:128920126259844867 FileName:assembly
FileAttr:32 CreateTime:129347621440351158 FileName:bfsvc.exe
FileAttr:22 CreateTime:129470486623450000 FileName:BitLockerDiscoveryVolumeContents
FileAttr:16 CreateTime:128920126264212839 FileName:Boot
FileAttr:36 CreateTime:128920210577598636 FileName:bootstat.dat
FileAttr:16 CreateTime:128920126265148833 FileName:Branding
FileAttr:16 CreateTime:129470486623450000 FileName:CSC
FileAttr:16 CreateTime:128920126265148833 FileName:Cursors
FileAttr:16 CreateTime:128920196613080734 FileName:debug
FileAttr:16 CreateTime:128920207509229246 FileName:diagnostics
FileAttr:16 CreateTime:129470481626887500 FileName:DigitalLocker
FileAttr:16 CreateTime:128920126265616830 FileName:Downloaded Program Files
FileAttr:16 CreateTime:129470486624075000 FileName:ehome
FileAttr:16 CreateTime:129470481626575000 FileName:en-US
FileAttr:32 CreateTime:129347621604931447 FileName:explorer.exe
FileAttr:21 CreateTime:128920126265616830 FileName:Fonts
FileAttr:32 CreateTime:128920003782689982 FileName:fveupdate.exe
```

## ps Plugini

Bu funksiyada Native Api əsasında prosesləri enum edir.

Plugini yükləmək üçün ilk öncə

```
from plugins.ps import *
```

```
>>> a = init()
```

```
>>> a
```

```
<ctypes.c_char_Array_56224 object at 0x019B9D50>
```

```
>>> enum(a,len(a))
```

```
<generator object enum at 0x01A0E5F8>
```

```
>>> for k in enum(a,len(a)):
```

```
...     k
```

```
...
```

Geriyə PROCESS\_INFORMATION\_BLOCK Strukturunu qaytarır.

- ('NextEntryOffset',c\_ulong),
- ('NumberOfThreads',c\_ulong),
- ('WorkingSetPrivateSize',c\_ulonglong),
- ('HardFaultCount',c\_ulong),
- ('NumberOfThreadsHighWaterMarks',c\_ulong),
- ('CycleTime',c\_ulonglong),
- ('CreateTime',FILETIME),
- ('UserTime',FILETIME),
- ('KernelTime',FILETIME),
- ('image',UNICODE\_STRING),
- ('BasePriority',c\_long),

- ('uniquid',c\_void\_p),
- ('parentid',c\_void\_p),
- ('HandleCount',c\_ulong),
- ('SessionID',c\_ulong),
- ('UniqueProcessKey',c\_ulonglong),
- ('PeakVirtualSize',c\_size\_t),
- ('VirtualSize',c\_size\_t),
- ('PageFaultCount',c\_ulong),
- ('PeakWorkingSetSize',c\_size\_t),
- ('WorkingSetSize',c\_size\_t),
- ('QuotaPeakPagedPoolUsage',c\_size\_t),
- ('QuotaPagedPoolUsage',c\_size\_t),
- ('QuotaPeakNonPagedPoolUsage',c\_size\_t),
- ('QuotaNonPagedPoolUsage',c\_size\_t),
- ('PageFileUsage',c\_size\_t),
- ('PeakPageFileUsage',c\_size\_t),
- ('PrivatePageCount',c\_size\_t),
- ('ReadOperationCount',c\_ulonglong),
- ('WriteOperationCount',c\_ulonglong),
- ('OtherOperationCount',c\_ulonglong),
- ('ReadTransferCount',c\_ulonglong),
- ('WriteTransferCount',c\_ulonglong),
- ('OtherTransferCount',c\_ulonglong),
- ('th',SYSTEM\_THREAD\_INFO),



## Virustotal plugini.

Import etmək üçün

```
from plugins.virustotal import *
```

vt adında əsas sinif mövcuddur.

SİNİF url,hash,domain adında 3 metodu özündə saxlayır.

domain-domainləri yoxlamaq üçün.

url-url adresini yoxlamaq üçün.

hash-hash summaları yoxlamaq üçün.

```
>>> from plugins.virustotal import *
>>> a = vt()
>>> a
<plugins.virustotal.vt object at 0x006948D0>
>>> dir(a)
['_class_', '_delattr_', '_dict_', '_dir_', '_doc_', '_eq_', '_format_', '_ge_', '_getattribute_', '_gt_', '_hash_', '_init_', '_le_', '_lt_', '_module_',
'_ne_', '_new_', '_reduce_', '_reduce_ex_', '_repr_', '_setattr_', '_sizeof_', '_str_', '_subclasshook_', '_weakref_', 'apikey', 'domain', 'hash', 'url']
>>> a.domain("027.ru")
{'BitDefender category': 'parked', 'whois': 'domain: 027.RU\\nnserver: ns1.masterhost.ru.\\nnserver: ns2.masterhost.ru.\\nnserver: ns.masterhost.ru.\\nstate:
REGISTERED, DELEGATED, VERIFIED\\nperson: Private Person\\nregistrar: RU-CENTER-RU\\nadmin-contact: https://www.nic.ru/whois\\ncreated: 2005.12.09\\npaid-till:
2015.12.09\\nfree-date: 2016.01.09\\nsource: TCI", "whois_timestamp": 1417711541.92504, "detected_downloaded_samples": [{"date": "2013-06-20 18:51:30", "positives": 2, "to
tal": 46, "sha256": "cd8553d9b24574467f381d13c7e0e1eb1e58d677b9484bd05b9c690377813e54"}], "response_code": 1, "verbose_msg": "Domain found in dataset", "Websense ThreatSeeker category":
"malicious web sites", "resolutions": [{"last_resolved": "2013-05-03 00:00:00", "ip_address": "90.156.201.11"}, {"last_resolved": "2013-05-07 00:00:00", "ip_address": "90.156.201.14"},
{"last_resolved": "2013-04-01 00:00:00", "ip_address": "90.156.201.27"}, {"last_resolved": "2013-05-01 00:00:00", "ip_address": "90.156.201.71"}, {"last_resolved": "2013-06-20 00:00:00",
"ip_address": "90.156.201.97"}], "detected_urls": [{"url": "http://027.ru/", "positives": 2, "total": 61, "scan_date": "2015-01-02 04:19:50"}], "categories": ["parked", "malicious we
b sites"]}]
>>>
```

## Script dili.

MAK-ın özünə məxsus script dili mövcuddur. Script dili hələlik yalnız təmizlik əməliyyatı aparmaq üçün nəzərdə tutulub.

Əsasən bəzi vacib əməliyyatları avtomatikləşdirmək üçün nəzərdə tutulub.

Sintaksis quruluşu belədir.

➤ Əmr("obyekt");

Əmrlər.

- delete\_file
- process\_kill\_pid
- process\_kill\_name
- qua
- debug
- reboot\_delete
- reg\_delete\_key
- reg\_delete\_value

Nümunələr.

```
delete_file("c:\\Windows\\worm.exe");  
process_kill_pid("2155");  
process_kill_name("malware.exe");  
qua("c:\\Windows\\worm.exe");  
reboot_delete("c:\\Windows\\worm.exe");  
reg_delete_key("HKEY_LOCAL_MACHINE\\SOFTWARE\\CBSTEST");  
reg_delete_value("HKEY_LOCAL_MACHINE\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run","Worm");
```

Script –ti istənilən mətn editoru ilə yazı bilərsiniz. Yaddaşa verdiyiniz zaman faylın formatını .mscr olaraq verməlisiz.



### **Logların paylaşılması.**

MAK sistemində yaradılan logların başqaları tərəfindən paylaşılması üçün webserv.py(pyc) –dan istifadə olunur.

Məhz bu modulun köməklili ilə logları uzaqdan müşahidə edə bilərsiniz.

İlk öncə təlimatın əvvəlində qeyd edilmiş sistemin yüklənməsi və sazlamalar bölməsində qeyd etdiyimiz sazlamaları edin sonra webserv.py(pyc) faylını işə salın.

Və artıq logları uzaqdan müşahidə edə bilərsiniz.

Aşağıda ki, funksiyalar heç bir sinifə aid deyil. Bu funksiyalar istifadəçinin rahatlığını təmin etmək üçün nəzərdə tutulub.

`select` funksiyası.

Bu funksiya istifadəçilərin rahat bir şəkildə faylları seçmələri üçün dialog pəncərəsini aktiv edir.

`cls` funksiyası.

Bu funksiya interaktiv shell pəncərəsini təmizləyir.

`terminate_all` funksiyası.

Bu funksiya parametr olaraq list obyekt qəbul edir siyahıda olan PID(Process ID)-ləri dayandırmaq üçün nəzərdə tutulub.

*Sonda qeyd etmək istəyirik ki, bu Malware Analizi üçün istifadəyə verdiyimiz ilk sistem-alətlər toplusudur. Yaxın gələcəkdə ümumi istifadə üçün daha avtomatizə edilmiş alətlər və program təminatlarını xidmətinizə verəcəyik.*

*Bu sistemin hazırlanmasındakı əsas məqsədlərdən biri isə bu sahədəki mütəxəssislər üçün baza alətlər panelini yaradaraq onların bu sahədəki bilik və bacarıqlarını artırmaqda onlara yardımçı olmaqdır.*

*Sistem çətin görünsədə istifadə etdikcə daha anlaşıqlı və rahat olduğunu anlayacaqsınız. Sistemdən istifadə sizin malware analizində nə qədər çox tərəfli düşünməyinizi, zərərvericiləri nə qədər yaxşı anlamağınızı və analiz etmə qabiliyyətinizin nə dərəcə də effektiv olduğunu da ortaya çıxarmış olacaq.*