Tilon/SpyEye2 intelligence report

Tilon, son of Silon, or…
SpyEye2 evolution of SpyEye?

The malware family commonly known as Tilon has been around for several years now. While several public analysis reports have described the malware; no one has thus far linked it with the well-known SpyEye malware family. In light of the recent news of the guilty plea of SpyEye distributor Gribodemon we revisit the Tilon malware family. We give a detailed analysis of similarities to SpyEye and also place Tilon and SpyEye into a wider context of the digital underground.

The original name Tilon was chosen due to the similarities with Silon. This was merely true for the outer layer of the malware, the so called loader. A better name probably was SpyEye2, as the functional part of the malware is sourced from SpyEye. The team behind its creation was similar, however reinforced with at least one better skilled programmer.

The decline in Tilon/SpyEye2 activity after the arrest of Gribodemon was evident, the development however continued and the fraudulent activities did not stop. Finally after nearly a year of declining usage, it seems we might have come to the real end of the SpyEye era, or will the team behind SpyEye2 continue and start working on getting new customers?

Read all the details in this intelligence report.

Fox-IT InTELL

SpyEye

SpyEye is one of the most discussed malware families in recent years. It quickly rose to become the kit-based financial malware of choice for many groups involved in online banking and online credit card fraud. Research from Fox-IT InTELL shows that the group behind SpyEye is also responsible for another key financial malware family, Tilon, that has wreaked havoc in Europe. With the 2013 arrest of the SpyEye author, the security community may have removed a player with even more involvement in the financial malware underground than previously thought.

Background

SpyEye's rise to fame was by no means apparent from the start, because although initially sold for a few hundred USD by its author, Gribodemon, in many cases it crashed on the systems where it was installed. For this reason, Fox-IT InTELL was initially hesitant to classify it as a major threat. However, during 2010, with continued support of its development and user community and an increasing customer base, SpyEye slowly gained in popularity and became a serious threat.

A major development and opportunity during SpyEye's evolution came when ZeuS left the market, with sales as well as support no longer offered by its author. It was announced that ZeuS support would be picked up by the SpyEye team, led by author Gribodemon, but this never materialized. Within the security industry speculation loomed about the emergence of a super trojan to be named SpyZeuS, and researchers went out of their way to find proof of this development. To date, this has never materialized.

Shake-up in the criminal underground

The reality of events and motivation behind the change in the online threat landscape was likely very different: Slavik, author of ZeuS, was involved in several projects and one of the most profitable was an operation that executed very large heists using corporate treasury account takeovers and different methods to get the money out of these (sometimes multi-million USD) accounts.

The business of supporting the kit malware ZeuS, offering support and dealing with countless user support questions, combined with large-scale piracy of his product likely led him to willingly give up this (otherwise profitable) business of selling a product with a near-perfect reputation and very high underground market value.

We see the sudden disappearance of ZeuS and the emergence of SpyEye as the prominent family of malware as a cunning plot of Slavik to divert the significant attention he had gained over the years, from both Law Enforcement and security researchers, onto another person: Gribodemon. Successfully it would seem, as whereas Slavik has never been arrested and is still the leader of a very successful criminal gang, Gribodemon is just beginning what will be a lengthy jail sentence.

Tilon

An interesting part of the story emerges when we examine the other activities of Gribodemon outside of SpyEye development. Little is publicly known about his other activities, until now: Our findings show that after the last release of SpyEye 1.3.48, in October 2011, the SpyEye team started a side project, developing a private trojan platform for rent. Such a business model would attract much less attention than a more widely-available kit malware and doesn't require follow-on support hassles encountered when selling a mass-marketed kit option.

It turns out this private trojan is the well-known managed malware family Tilon. Tilon was first published on in August 2012 by Trusteer and classified as malware based on Silon, but this is only partially true. Tilon is actually SpyEye2.

Tilon has been an active malware family in the wild from 2012-2014, but recent activity levels have been quite low and currently there appear to be no active C&C servers. Its activity has been focused mostly on Europe with the longest-known campaigns targeting Italian banks with various attacks, including using the latest versions of mobile malware available at the time, such as Perkele, used to circumvent mTAN.

The team involved in its development and operation was the same as the team behind SpyEye. There is much circumstantial evidence to support this theory. Examples include SpyEye customers who migrated to Tilon and also a sharp decline in activity of Tilon after the arrest of Gribodemon.

Technical Evidence

If one knows where to look, there is much evidence to show that Tilon (from this point referred to as SpyEye2) is based on the SpyEye source code.

This relationship is not immediately apparent, because the code base has been overhauled. It is evident that the code has been modified by at least one new programmer, resulting in parts of the code base that are completely rewritten, in a more elegant, and robust way. SpyEye2 now comes with different modules for both the CPU platform (32 and 64 bit), but also different versions for Windows XP to Windows 8.

Diving into the code, we can see that the SpyEye collector code was reused and some SpyEye specific features were carried forward. In some cases the author opted to rewrite these functions, but in the majority of cases, these functions were cleaned up but retain most of their original structure. A key observation is the use of the same version of the LZO library from the SpyEye tree. If SpyEye2 was indeed an independently-developed trojan, it would be logical to opt for the newer, 2010 version of the LZO library.

Looking at the backend of SpyEye2, much has changed. There is a single backend system strongly resembling the original SpyEye RDP backconnect daemon and also containg a lot of code from the SpyEye collector, but using the HTTP protocol. The server side component is called "dae" (short for daemon, a common name for a Unix service, which was also used for the RDP backconnect component of SpyEye), and combines bot control, log data, RDP and socks functionality and webinject configuration management in a single platform.

There can be multiple instances of "dae" which can talk to the same backend database and in front of any "dae" instance the operator can place multiple layers of HTTP proxies or fast flux HTTP proxy networks to make the network more resilient against takedown.

The management of SpyEye2 is done through a single, unified interface, which has been completely redesigned but still contains a few of the unique features of the original SpyEye. While the original SpyEye was heavily branded to

improve product marketing, SpyEye2 is unbranded like many other banking malware services, as they do not rely on public marketing but instead on trusted introductions of new and reliable customers.

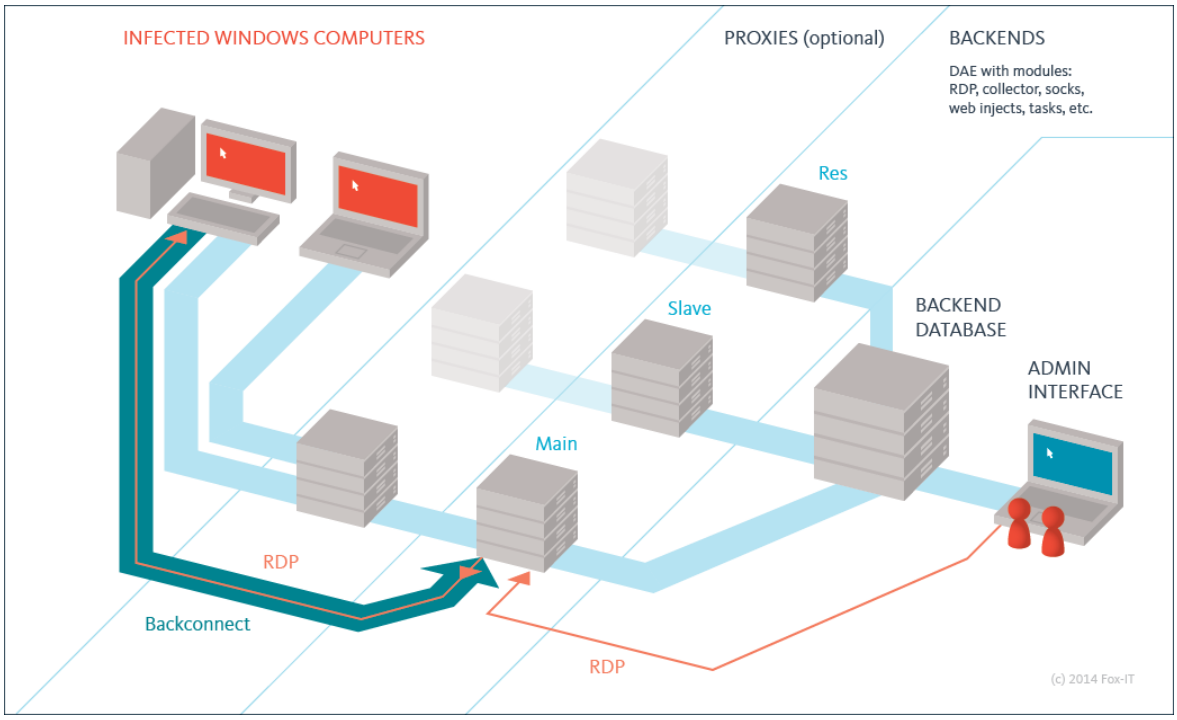


Figure 1. Overview of SpyEye2 infrastructure

An interesting and slightly funny feature of some of the earlier SpyEye2 versions is the ability to remove the original SpyEye malware. No other malware families are checked for removal. Early versions of the original SpyEye were likewise equipped with a feature to remove older versions of ZeuS installed on the infected system.

Looking specifically at the SpyEye2 internals, it has become a much better and robust trojan platform than its market competitors, both in terms of features and stability, but also in security, an area in which the original SpyEye was lacking.

The loader portion of Tilon is sourced from Silon, but this is where the similarity ends. As shown above and further illustrated in the Appendices, the body (i.e., functional portion) of Tilon was actually based on SpyEye. This is very compelling evidence because it implies that the SpyEye2 developers had access to the core SpyEye source code.

In contrast, other malware families that have incorporated features from SpyEye, have only copied supporting modules, such as the RDP or Socks backconnect functionality. One such trojan is Tinba v2 which contains functionality partially based on SpyEye backconnect components.

Conclusions

The criminal underground is a fascinating and complex subject to write about. It takes years of experience and knowledge to correctly assess underground activity and turn it into threat intelligence, it also requires expertise in a variety of areas to discover all the insights to correctly classify a threat.

We have seen that after the arrest of the SpyEye author, SpyEye2's activity level has dropped dramatically, even though it has still been extended and updated to support the latest versions of browsers up to December 2013.

Fox-IT views arrests like Gribodemon and other key figures in the underground economy such as Paunch, the author of the popular Blackhole Exploit Kit, as the key to decreasing the worldwide activity around online crime. While other actors can replace their knowledge, these actors are an important lynchpin interconnecting underground trust relations. Breaking these trust networks splits the criminal underground into isolated islands.

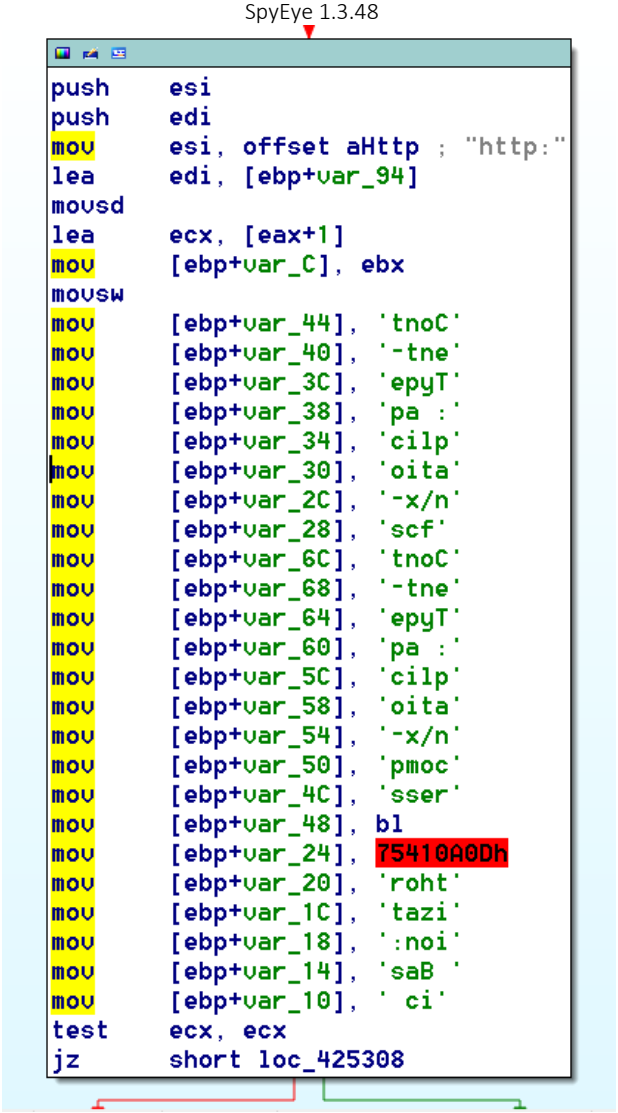
Will we see more of SpyEye2 in the future? It's impossible to predict the future, however it is likely that the group will continue with their activity in some form. We are currently observing a move from certain groups that were focused on online banking fraud to other methods of fraud such as ransomware, point-of-sale (POS) malware and general credit card fraud, click fraud and even more frequently digital currency account or wallet hijacking and the mining of cryptographic currency (e.g., Bitcoins).

Appendix I – Similarities in SpyEye 1.3.48 and SpyEye2

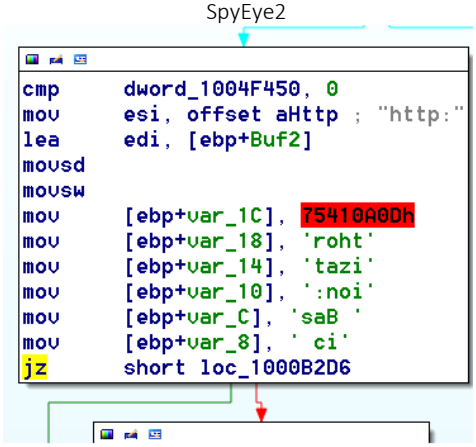
Due to the fact that SpyEye 1.3.48 and SpyEye2 are using different compiler versions and compiler options and a lot of rewriting of code there are quite some changes in how certain functions appear when disassembling. However the basic underlying function and content, often typical for SpyEye, remains the same.

Looking at some of the functions which are unique to SpyEye, the exact same code can be found in SpyEye2 (Content-Type: application/x-fcs, Content-Type: application/x-compress, Authorization: Basic):

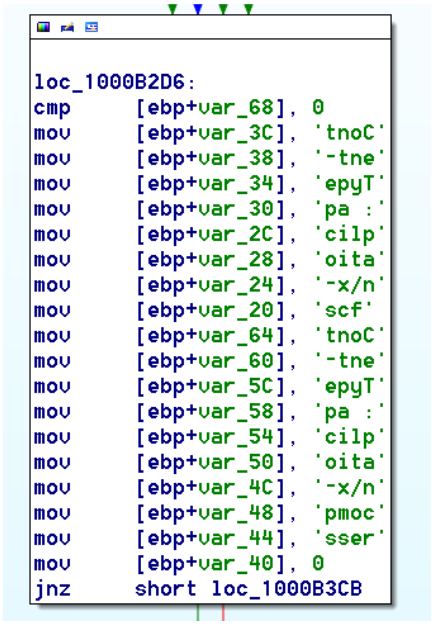
SpyEye 1.3.48



SpyEye2



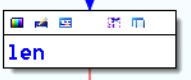
...



Here is an example of a function related with the SpyEye Collector code, while you can spot the differences, the similarity is obvious.

SpyEye 1.3.48

```
push    eax
push    offset aProcess_name ; "process_r
push    edi
lea     ecx, [ebp+var_8C]
call    sub_41CFC2
mov     eax, [ebp+arg_0]
lea     esi, [eax+1]
```



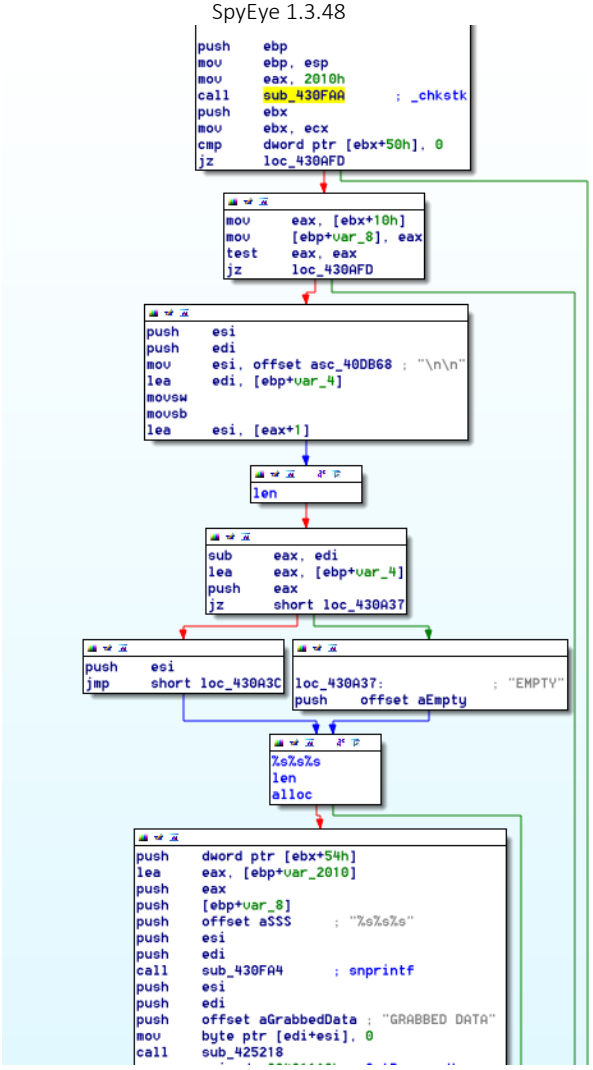

```
push    [ebp+arg_0]
sub     eax, esi
push    eax
push    offset aHooked_func ; "hooked_fu
push    edi
lea     ecx, [ebp+var_8C]
call    sub_41CFC2
push    [ebp+arg_4]
lea     ecx, [ebp+var_8C]
push    [ebp+arg_8]
push    offset aFunc_data ; "func_data"
push    edi
call    sub_41CFC2
mov     eax, ds:0C433A30h
mov     esi, 0C42EC10h
push    esi
add     eax, eax
push    eax
push    offset aKeys    ; "keys"
push    edi
lea     ecx, [ebp+var_8C]
```

SpyEye2

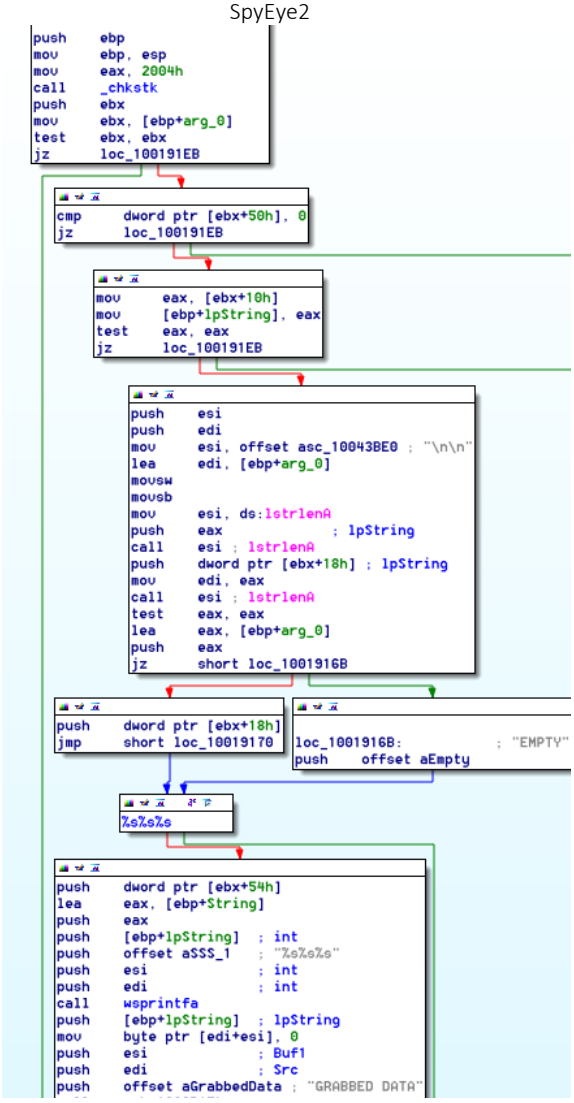
```
push    eax             ; dwBytes
push    offset aProcess_name ; "process_na
push    ebx             ; int
push    esi             ; int
call    sub_10006008
add     esp, 14h
push    [ebp+arg_0]     ; int
push    [ebp+arg_0]     ; lpString
call    edi ; lstrlenW
add     eax, eax
push    eax             ; dwBytes
push    offset aHooked_func ; "hooked_func
push    ebx             ; int
push    esi             ; int
call    sub_10006008
add     esp, 14h
push    [ebp+lpMem]     ; int
push    [ebp+lpMem]     ; lpString
call    edi ; lstrlenW
add     eax, eax
push    eax             ; dwBytes
push    offset aFunc_data ; "func_data"
push    ebx             ; int
push    esi             ; int
call    sub_10006008
add     esp, 14h
mov     edi, offset stru_10059198
push    edi             ; lpCriticalSecti
call    ds:EnterCriticalSection
mov     eax, dword_10053F64
push    offset word_10053F68 ; int
add     eax, eax
push    eax             ; dwBytes
push    offset aKeys    ; "keys"
push    ebx             ; int
```

Yet another function related to the SpyEye Collector:

SpyEye 1.3.48

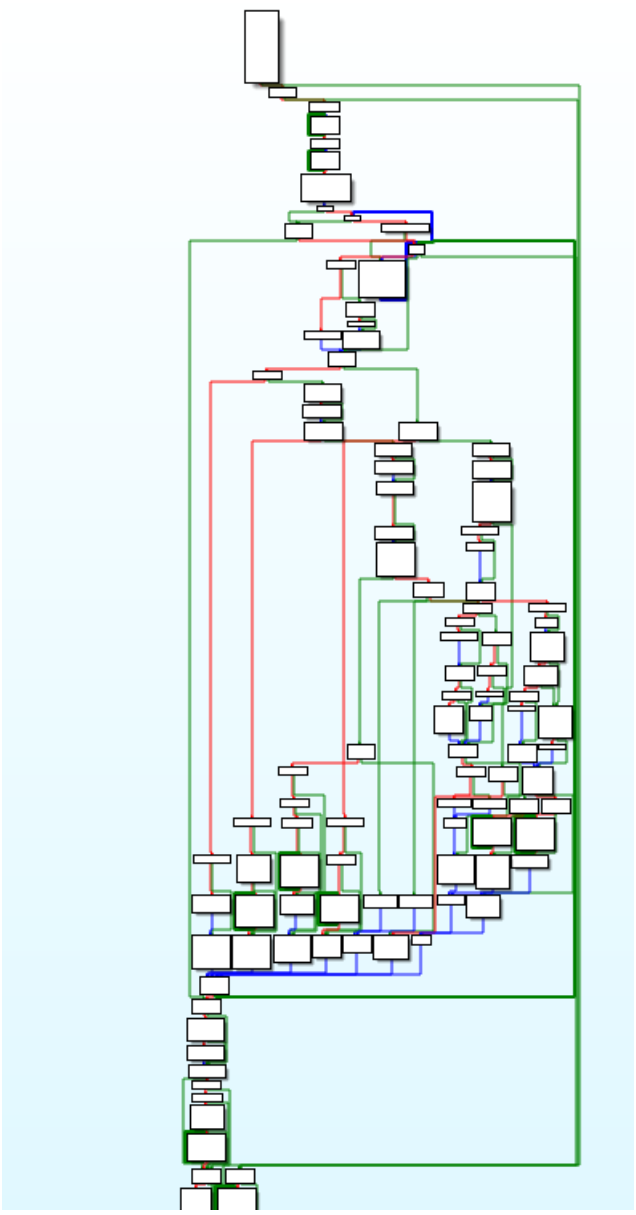


SpyEye2

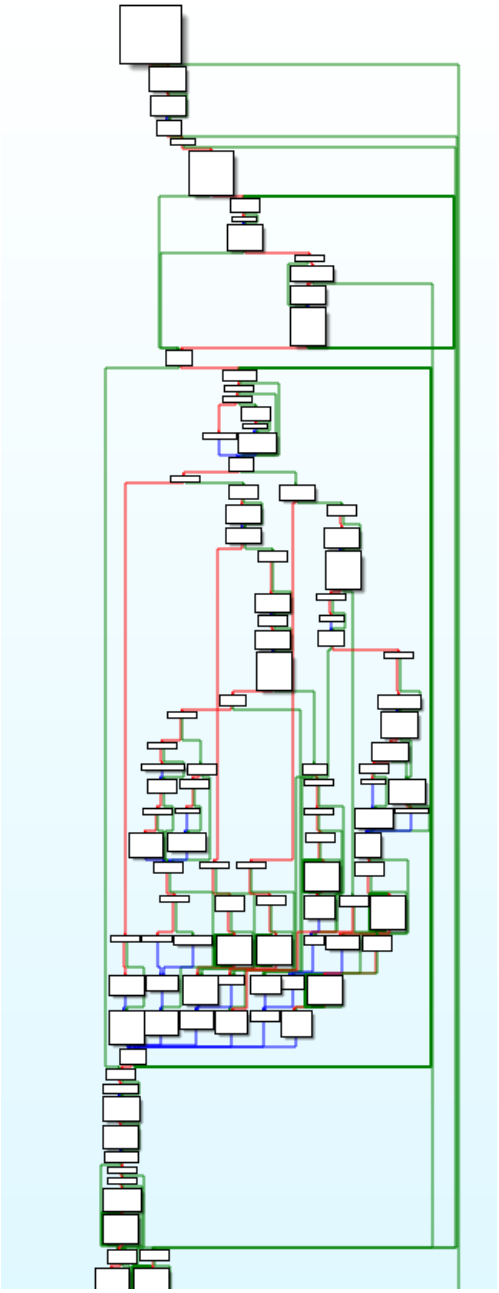


A function related with the handling of webinjects, looking at the whole function in graph mode, even without looking at the details and with an untrained eye you can probably note the similarities in the two functions:

SpyEye 1.3.48



SpyEye2



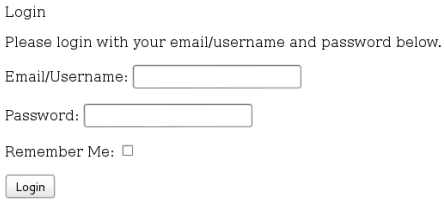
Looking at for example the settings which are changed in FireFox, these are done in an identical way in both the original SpyEye and the SpyEye2 version.

```
unicode 0, <\prefs.js>,0
db 'user_pref("browser.safebrowsing.enabled", false);',0Dh,0Ah ; DATA XREF:
db 'user_pref("browser.safebrowsing.malware.enabled", false);',0Dh,0Ah
db 'user_pref("security.warn_entering_weak", false);',0Dh,0Ah
db 'user_pref("security.warn_entering_weak.show_once", false);',0Dh,0Ah
db 'user_pref("security.warn_viewing_mixed", false);',0Dh,0Ah
db 'user_pref("security.warn_viewing_mixed.show_once", false);',0Dh,0Ah
db 'user_pref("privacy.clearOnShutdown.cookies", false);',0Dh,0Ah
db 'user_pref("privacy.clearOnShutdown.sessions", false);',0Dh,0Ah
db 'user_pref("network.http.spdy.enabled", false);',0Dh,0Ah,0
```

The only difference in SpyEye2 is the addition of the SPDY specific option, which was introduced to mainstream FireFox in March of 2012.

Appendix II – Peek into backend infrastructure

Login screen for users of SpyEye2:



The backend of SpyEye2 has a large number of options, with an obvious overview of infected systems and access to logdata. The grabbed data from victims such as HTTPS POST requests containing passwords are called “Sausages”, the reason for this choice of word remains a mystery to us. Other specific stolen information is available separately such as stored certificates, keys and FTP credentials.

The accounts patterns is a special feature which specifies patterns on how to extract high value username and password combination from the earlier mentioned “Sausages” data, it is therefore called “Sausage patterns”. Yes, we also had a laugh when seeing this first and when writing it down.

Looking at the database structure of the reports/sausages table, you can see many similarities, some changes are logical, for example the botid in SpyEye (bot_guid) is a string (consisting of the windows version, hostname and a unique hash), while in SpyEye2 it is a numeric ID.

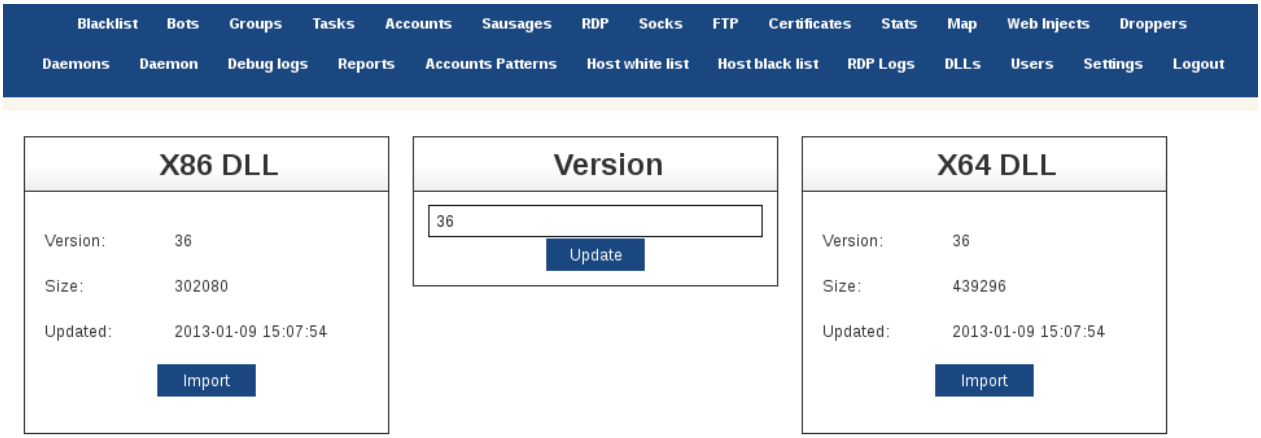
SpyEye

```
CREATE TABLE IF NOT EXISTS `rep2_` (
 `id` bigint(20) unsigned NOT NULL
auto_increment,
 `bot_guid` varchar(40) NOT NULL,
 `process_name` varchar(270) NOT NULL,
 `hooked_func` varchar(100) NOT NULL,
 `url` varchar(2112) NOT NULL,
 `func_data` varchar(333000) NOT NULL,
 `keys` varchar(10000) character set ucs2
collate ucs2_bin default NULL,
 `date_rep` datetime NOT NULL,
 PRIMARY KEY  (`id`)
 ) ENGINE=MyISAM ;
```

SpyEye2

```
CREATE TABLE IF NOT EXISTS `sausages` (
        `id` bigint(20) unsigned NOT NULL
AUTO_INCREMENT,
        `botid` bigint(21) unsigned NOT
NULL,
        `uid` int(11) unsigned DEFAULT '0',
        `process_name` varchar(270)
character set ucs2 collate ucs2_bin DEFAULT
NULL,
        `hooked_func` varchar(100)
character set ucs2 collate ucs2_bin DEFAULT
NULL,
        `url` varchar(2112) NOT NULL,
        `func_data` varchar(65535)
character set ucs2 collate ucs2_bin NOT
NULL,
        `keys` varchar(10000) character set
ucs2 collate ucs2_bin DEFAULT NULL,
        `gkeys` varchar(65535) character
set ucs2 collate ucs2_bin DEFAULT NULL,
        `time` int(11) unsigned NOT NULL,
        `processed` tinyint(1) DEFAULT '0',
PRIMARY KEY (`id`),
KEY `botid` (`botid`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8
AUTO_INCREMENT=1;
```

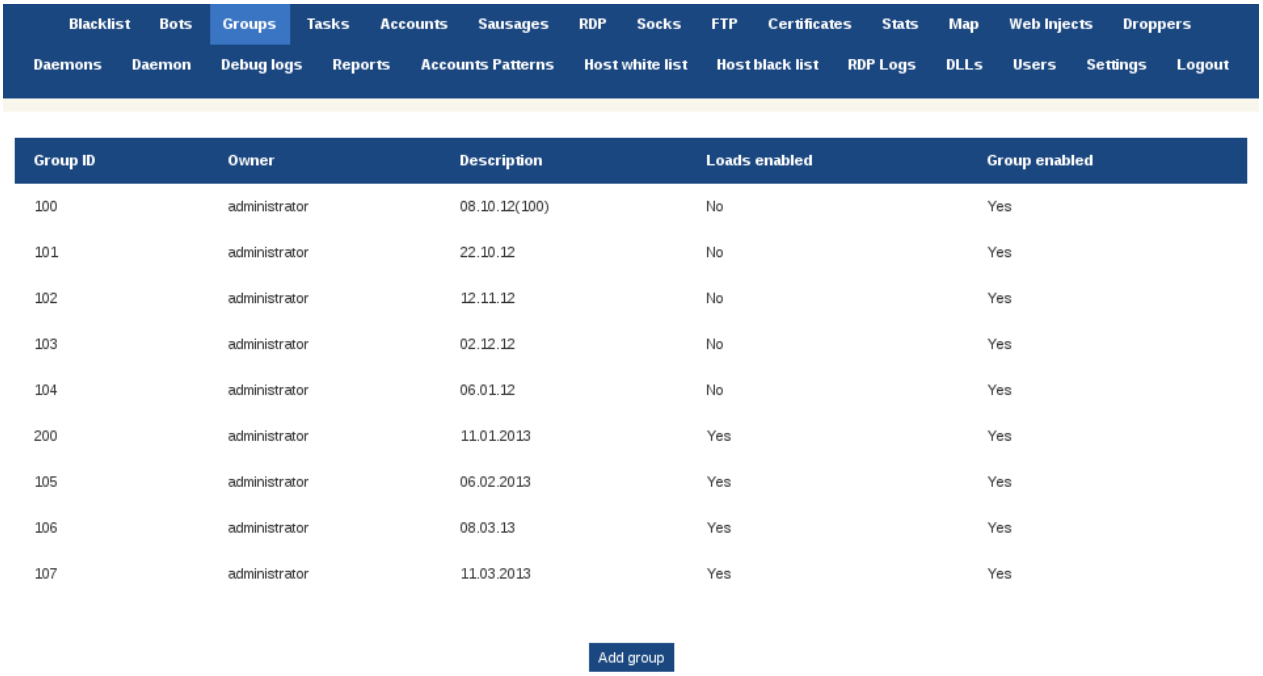
SpyEye2 initially just had a separate version for 32 and 64 bit systems, but later versions were actually equipped with multiple variations for each major Microsoft Windows platform.



In the second half of 2013 the different versions of SpyEye2 were available for :

- Windows XP – 32bit
- Windows Vista – 32bit
- Windows Vista – 64bit
- Windows 7 – 32bit
- Windows 7 – 64bit
- Windows 8 – 32bit
- Windows 8 – 64bit

Looking in the specific Groups tab, are the botnets within an instance of SpyEye2, also often known as sub-botnets. These can be used to distinguish infection campaigns or sets of different bots from varying countries.

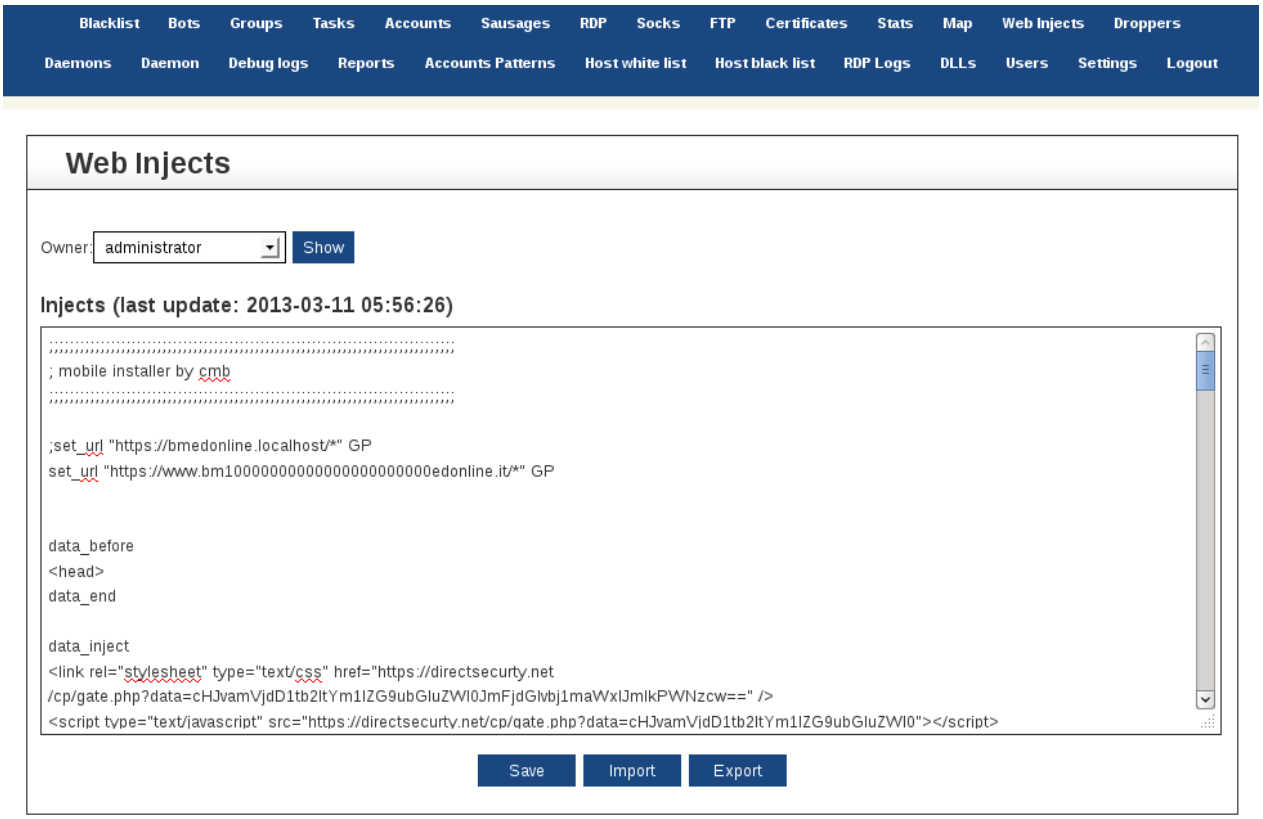


Group ID	Owner	Description	Loads enabled	Group enabled
100	administrator	08.10.12(100)	No	Yes
101	administrator	22.10.12	No	Yes
102	administrator	12.11.12	No	Yes
103	administrator	02.12.12	No	Yes
104	administrator	06.01.12	No	Yes
200	administrator	11.01.2013	Yes	Yes
105	administrator	06.02.2013	Yes	Yes
106	administrator	08.03.13	Yes	Yes
107	administrator	11.03.2013	Yes	Yes

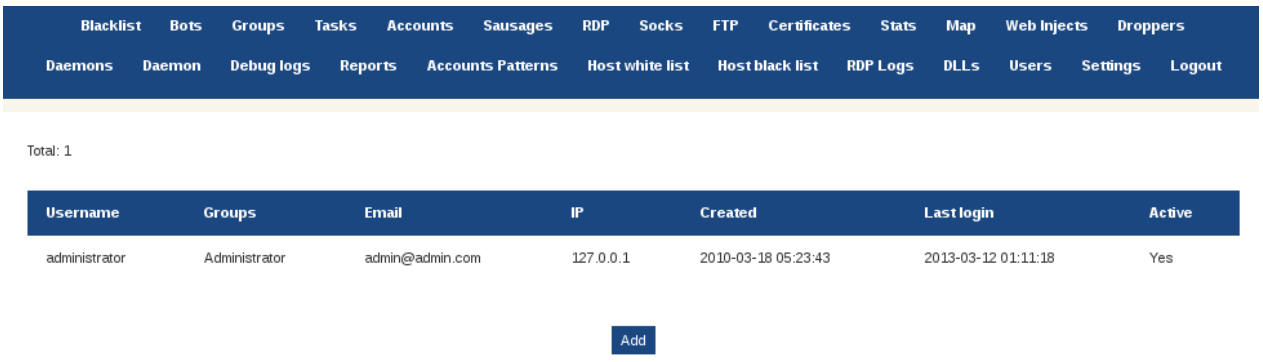
Add group

An interesting feature that can be observed here is the Loads Enabled feature, not allowing new bots to be added to a sub-botnet after an infection campaign is finished, effectively stopping researchers to fully research the additional components of the malware.

A built in editor for webinject code of SpyEye2 backend, showing in this case an attack against an Italian bank using the Injeria webinject mobile backend, social engineering victims to install the Perkele mobile malware. This dynamic editing makes rolling out webinject changes to infected systems very easy and flexible.

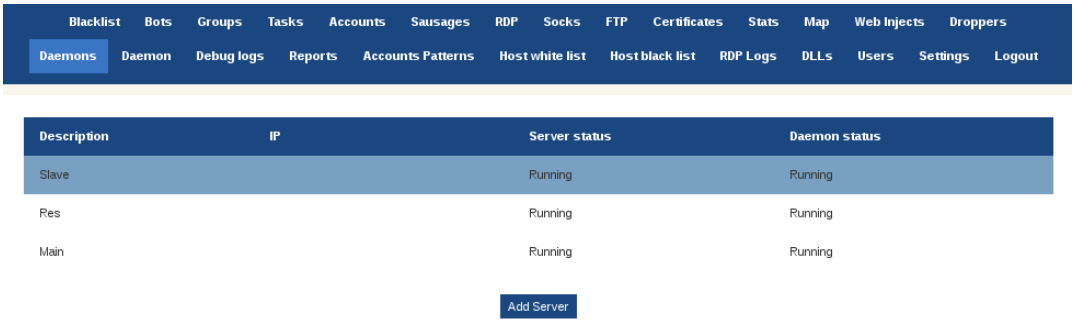


Overview of users allowed to login to the backend of SpyEye2:

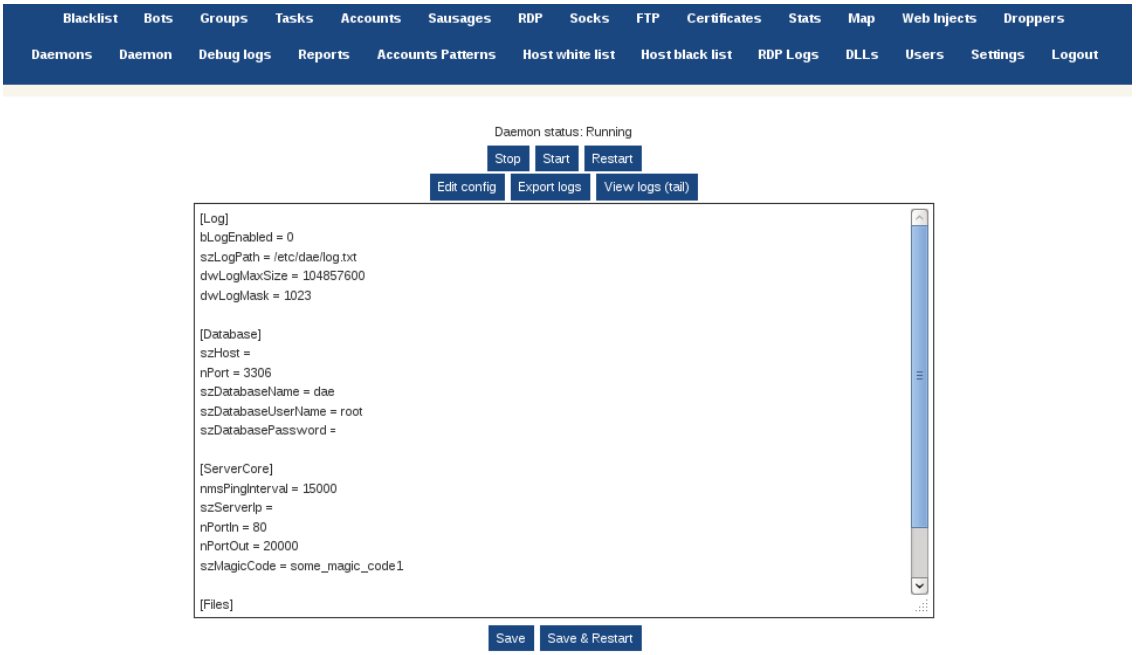


The creation date of the default administrator account has nothing to do with the start of SpyEye2 activity, but is merely a result of using the Codeigniter framework which contains a default administrator user.

Overview of the Daemons, which are the “dae” server-side component mentioned earlier:



Editing the configuration of a specific “dae” component, that also specifies “nPortOut” which is used to bind backconnect endpoints.

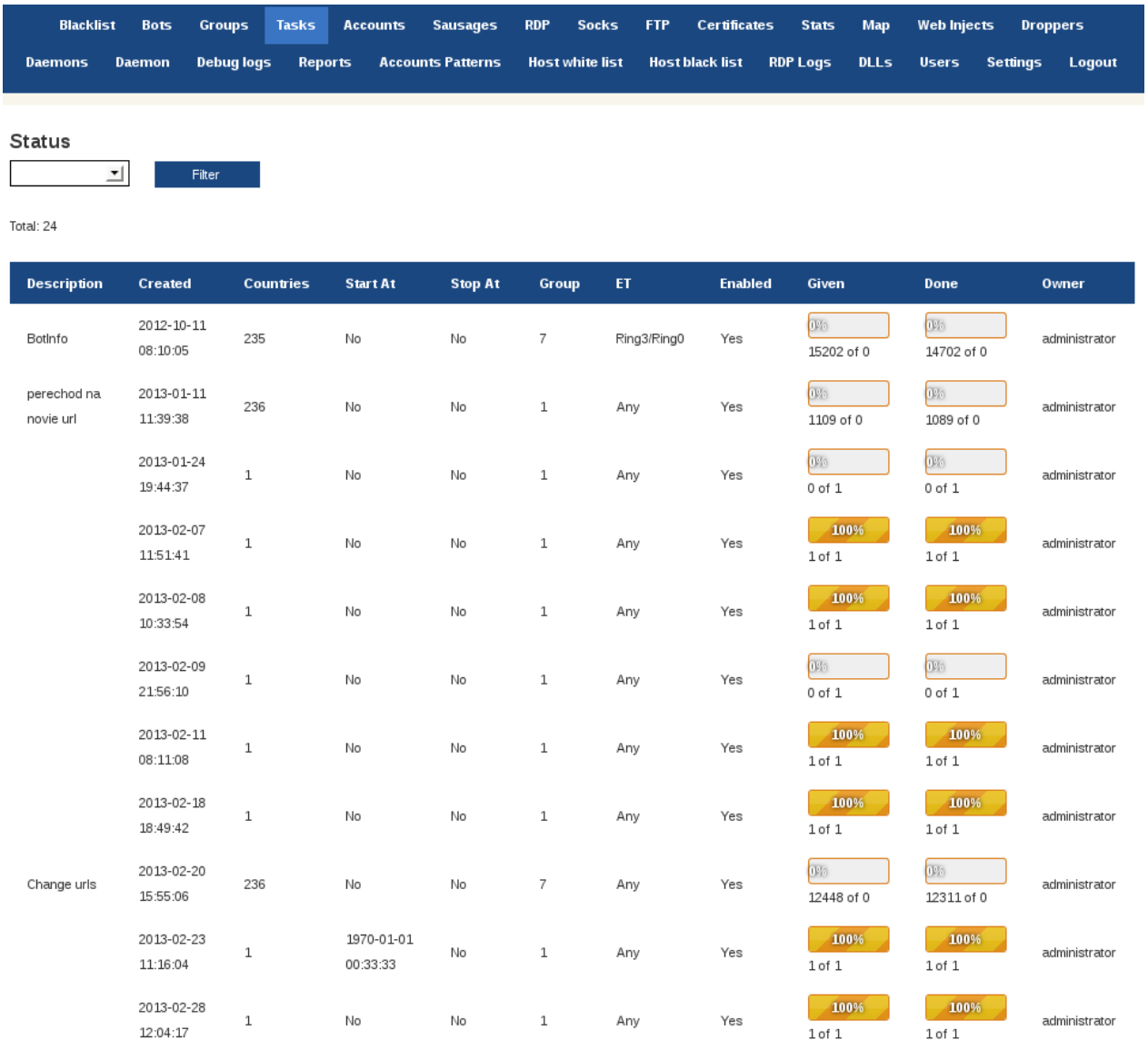


Looking at the old SpyEye RDP daemon the configuration looks very familiar:

```
[options]                                   cfg_file_log_enabled = 1
mysql_host = localhost                      cfg_file_log = /etc/dae/main.log
mysql_port = 3306                           cfg_file_log_maxsize = 1048576
mysql_db = duffy
mysql_user = root                           cfg_file_blacklist =
mysql_pass = 12345                          /etc/dae/blacklist.bin
mysql_table_rdp = rdps2                     cfg_ip_address = 0.0.0.0
mysql_table_logs = nlogs2
mysql_ntries_to_repeat = 5                  cfg_rdp_port_in = 30000
mysql_intervalof_repeat = 30                cfg_rdp_port_out = 30010
                                            cfg_nmax_bots = 450
ping_interval = 30000
ping_db_update_multiplier = 10
```

```
magic_code = some_magic_code
```

An overview of the running tasks, showing some of the detailed options that are available when executing a task, with country specific settings and detailed progress indicators.



Appendix III – List of hashes

SpyEye2 and modules:

```
210ac7d65bc09c948670fa57daa89113
2b2ad42b9dcccc6aa985937450a00f4b
32cd78e6d8ccf7c694dd0b41923371a9
55739027848544ec2adc92423a008868
662d32a69754fee7b3e8047eafb5dc98
6785edf8afa560df1e5168aab121b5d7
6def387d177a705f36f87afa8c093cfe
712cf813b3c7c0c58841924ad2dd5734
8bd515b0cd5b8155787748b3edeaa80b
8e7c7dde842223bfa7d09680f9b74f5c
8f749a6206d5e4bd273bb3eaf146ff0b
90613574a81e9cf31bc51da8528f1be4
974c7e9cccda1d1596c6116885148a39
a04aaafa32a6be7f7c461a5335ca8dc5
a262ff16dc300d77a9afc84ed4f63c48
af39380c571eb235b6e4ecd0c60c6f00
b14a734490d7d91121b09c127b74ec89
bd2cd8647878de10ccec254c24f35cc8
ccfd1cae4ad02e5df3a185ed7b28ca7a
d6fe7ffc4b6d0622eb7c9ffede32dff3
e117a9eadf56bbf51cea310c6f818bac
e3e86075b58709dbb1ea85faa733d1ba
ecff29d12bcc050ec8b4411ca1b4212c
f037d979111645ee41993b0dbf7b3a68
f82ee6d58f134ba49da029859e5dd0a5
fb796eba6fc8842ec775eb456fb0e414
fce16f572a3587e89fa0a3861dfc5f68
```

Dae and modules (SpyEye2 server components):

```
15d4010198cf049a47d4d8dd10a984af
1a749b1ef47894ab8a5df745aa62f62a
1f3bebddb158e1a4d9f41175c08d74f0
34242b3bb5314d212826f4d3ea1bb3f3
463b3cdf13f3aae68a40b42846ac63ed
826ab6487c405cdc13f8fb5da9537511
898c56150dd079cf9975d15af48ae225
a4e128daf7700360a7fdcf227f6df735
d6e6c784d9e9c942ceb2d2ecf5e517cb
```

SpyEye 1.3.48 unpacked (used for comparison):

```
9e3d385271ea97b489f0f13f82b3cdcd
```

RDP, FTP, Socks, SpyEye collector (original SpyEye server components):

```
0790cf56b64abab9c1480d01702a91a4
53a1662a7e3a14acc00a6d8e2b2b5b8d
5d2842fd36bd0123d9c2a843d42e0bb9
cd51bac98db504ce7eb0acd2efcadd15
```

Appendix IV – Public references

http://www.trusteer.com/blog/tilon-son-of-silon

http://news.techworld.com/security/3435936/police-arrest-london-man-in-connection-with-tilon-bank-trojan/

https://www.seculert.com/blog/2013/04/magic-persistent-threat.html

https://www.seculert.com/blog/2013/04/magic-malware-faq-and-iocs.html

http://rt.com/news/extradite-russian-national-panin-901/

http://www.fbi.gov/atlanta/press-releases/2014/cyber-criminal-pleads-guilty-to-developing-and-distributing-notorious-spyeye-malware

https://www.dhs.gov/sites/default/files/publications/nppd/ip/daily-report/dhs-daily-report-2014-01-30.pdf

http://krebsonsecurity.com/2014/01/feds-to-charge-alleged-spyeye-trojan-author/

http://www.darkreading.com/attacks-breaches/spyeye-creator-got-sloppy-then-got-nabbe/240165783