TheoryGroup

Where Theory meets Practice

Tracking Hackers on IRC

By: David Brumley [mailto:dbrumley@theorygroup.com]

Few hackers are motivated purely by knowledge, science, and curiosity. Hackers continue to break into systems long after they become familiar with the technology. Instead, many continue to hack simply because of the social status it brings. For many, hacking *is* a social activity. Hackers meet online to discuss the latest hacking tools, their hacking conquests, and their personal life. System administrators and security professionals must become familiar with the social culture of hackers dwell in to be truly effective.

Internet Relay Chat (IRC) has replaced electronic bulletin boards as the social mecca for internet addicts. Hackers are no expection. Cybersleuths must understand the jargon and tools used in this virtual society. By understanding IRC tools and jargon a cybersleuth can determine the real identity of a hackers from birthplace to current address and telephone number.

System logs help administrators and security professionals track down criminals. They are useful evidence that a crime has been commited, but not much else. System logs show how and where the electronic bits came from, but they don't show *who* sent them. To prosecute successfully you must not only show where the intruder came from, but who was physically using the keyboard at that particular time. IRC can be a tool for finding out.

For example, even with a full audit trail showing an intruder came from a particular account on a particular ISP, the most you can hope to obtain is billing information for the account. While sometimes sufficient, you still haven't show *who* was using the keyboard at the other end of the connection. The account you traced may have been stolen, set up with false billing information, or shared among several in a household. An IRC savvy administrator, however, may be able to determine the exact identity of the intruder. How? By listening to the hacker on IRC and reviewing configuration information on IRC tools left behind. A hacker bragging about compromising your host is also a full confession when logged. The IRC tools may be configured to always allow particular ISP connections, which may help in pinning down their

location. With a little deductive reasoning you can pin down who hacked your machine, what their name is, where they live, and even their favorite corner liquor store.

There are litterally dozens of IRC Networks. The most popular are DALnet, EFNet, and Undernet. Each IRC network is composed of hundreds, perhaps thousands, of channels where individuals with similar interests can chat real-time with each other. Channels are dynamic by nature. A channel is created the first time someone enters and destroyed when the last person leaves. The first person in a channel is also the channel operator, known as "chanops", or simply "ops". A channel operator is the super user for the channel: they can invite other users to the channel, set the topic, decide who can talk, and give or take operator status from others on the channel.

On some networks, such as DALnet and Undernet, channels can be registered after creation. Registration allows the creator to become a channel operator every time they log on to their channel. Registration assigns ownership of a channel.

Many IRC networks, including the ever popular EFNet, don't have channel registration. When you leave a channel, you leave all your privileges in that channel as well. You must be re-op'ed every time you join the channel. Hackers love dynamic networks like these because it allows them to take over channels. Hackers will force all legitimate users out of a channel until they are all that is left. When they are the only ones in the channel, they can op themselves. The primary method for forcing users off of a channel are Denial of Service (DoS) attacks. If the victims computer is swamped by a DoS, it will time out and disconnect from IRC.

Hackers who participate in these dynamic IRC networks have one primary goal: to keep operator status on the channels they frequent. To do this they must protect against others trying to take over their channel, rogue administrators randomly de-oping them, and deal with the inevitable denial of service attack. To solve these problems, hackers have come up with ingenious ways to create redundant connections from multiple hosts to and IRC network.

The simplest solution is to run multiple IRC clients, such as ircII or BitchX from several hosts. By running the clients under screen(1), a unix terminal multiplexer, they can detach IRC sessions into the background and reattach to them later. Each session corresponds to one nickname on the IRC network. If one host goes down, he can always re-attach to a running session on another machine. Since each nickname has operator status, the whole scheme is redundant. It is up to the intruder, however, to maintain daily every IRC session on every host - a very labor intesive activity.

IRC 'bots', short for robot, solve the problem of 'hands-on' administration. The purpose of a bot is to sit on IRC and monitor channels for events. In a very simple sense bots are only automated IRC clients. Running stand alone, a bot will automatically op friends (as specified in the configuration file), enforce bans for channel misuse, and provide some channel misuse control . The true power of bots, however, is their ability to link together to form "botnets". Each bot on a botnet serves as a redundant backup, automatically oping friends and other bots, enforcing channel bans, and ensuring a party line exists.

Each bot on the botnet is a node. The botnet administrator appoints a master node, with the rest becoming slaves. The master node is in charge of distributing botnet configuration information with each slave. After initial configuration, the botnet administrator need only change configuration information on the master. The master will then automatically take care of updating all the sub-nodes.

To add a bot to the network only a simple and static configuration file that specifies the master is needed. Once the new bot starts up, it will automatically contact the master and pull over the requisite configuration information to become a node on the botnet. The advantage to a hacker is enormous. For each new account or system compromise, the hacker need only upload the actual executable and a simple configuration file. Once started, the new bot automatically downloads all information including current lists of channels, friends, and users. The new bot will also then automatically update every time the configuration on the master is changed.

The most famous bot is "eggdrop", available at http://www.eggdrop.net. It serves as a good model for the typical bot. It's configuration file is divided up into three logical sections (sometimes in three separate files, sometimes merged into one): user information, channel information, and bot information.

Channel information can be recognized by the "channel add" TCL command. Following the channel name are a list of options to apply to that channel. A sample eggdrop channel file looks like:

```
channel add #myhacker {
        chanmode "+ismt"
        dont-idle-kick
        +userbans
        +protectops
}
```

The chanmode defines what mode the channel should be. In this particular example channel #myhacker is invite only (i), secret (s), moderated so that only channel operators can talk (m), and only channel operators can change topics (t).

The last three items define eggdrop configuration variables. Entries that begin with a plus (+) will enable options, entries that begin with a minus (-) will disable options. Entries will neither a plus or minus simply define a variable, i.e. make it true. In this particular example the bot will not kick idle users from #myhack, it will let user operators (as opposed to other bots on the botnet) ban people, and will automatically re-op de-oped users. For a full list of options, see the example configuration file that comes with the eggdrop distribution.

A user entry for an eggdrop bot can consist of four lines. The first line is always contains the nickname, password, and flags of the bot user. The remaining three lines all use the first two characters to identify the type of configuration information. Entries that start with a "-" list user identifier. To a bot, a user's identity is not their nickname, but the username@hostname.domain.zone where they are connecting from.

A line that begins with ":" is a botnet configuration entry. It lists HOSTNAME:PORT that the particular bot for that user will listen on. When two bots communicate they use the port on the host listed.

Lines that begin with "!!" or "." contain time stamp information on the user. Entries beginning with "!!" are the channel name and time stamp where the user was last seen by the bot. Entries with "." are the modification time of the entry itself. All times are kept in UNIX epoch format.

User files often contain dozens of bot users. If you've found an eggdrop configuration file on a compromised host, chances are most of the entries in the file are also compromised hosts or accounts. A quick note to the administrator of each domain explaining that you've found a hacker configuration file that references his domain is appropriate. You can also use the information in creating an MO (Modus Operundi) file for the hacker. People listed in the user file are often friends of the hacker (whom you may see in the future :) or alternate nicknames the hacker may be using.

Here's an example of eggdrop user file:

```
eleet     lypmjwfp2ee          fbs                          /0 0 0 0
-         *!eleet@*.elaine.Stanford.EDU,
*!eleet@*.myth.stanford.edu
:         firebird.stanford.edu:60000
```

```
!!          895178133 #stanford
.           {created 894412528}
```

Hackers often will not connect to IRC directly. By using a variety of hosts a hacker can subvert a ban, trick others into thinking he is someone else, or connect to an IRC server that limits connections. Most often, though, it is to hide his real IP address in case someone is watching them.

A "bounce" program reads from one port and writes to another, i.e. a proxy. The most famous bounce programs are BNC and WinGate. Both accept a TCP connection, connect to a destination, and then relay anything from the original connection to the destination. The primary legitimate use for WinGates are SOCKS and TCP proxys to the internet. Although WinGates can be configured to require a password, most are not. When a hacker has access to a wingate he can "bounce" through the wingate server to hide his tracks.

BNC, the word "bounce" with the vowels removed, are UNIX based proxy's designed primarily for "bouncing" IRC traffic. While a WinGate can proxy multiple ports, a BNC runs as a daemon listening to only one port. After accepting a connection, they too proxy information read on the original connection to a destination. In addition to simple proxying, the BNC configuration file allows for creating fake ident responses, virtual host configuration, and limiting the number of users who can use the bounce.

Since these processes run for extended periods of time, a hacker will often try to hide them from an administrator. If a hacker has superuser access and is skillful he can hide any process from any administrator. Luckily many hackers are sloppy or lazy. Often they will just change the name of the program do be something innocuous. A local favorite seems to be "pine". The hacker runs the process under the new name hoping that the administrator will not notice.

Because hackers are adept at hiding process names, you should always be aware of the network connections your host generates. netstat(1m) and lsof (http://vic.cc.purdue.edu/pub/tools/lsof_4.45_W.tar.gz) are good tools for monitoring local network connections. An administrator should also be wary of local processes, such as "./pine" or "./emacs" binding to unusual ports. It's a safe bet that pine doesn't listen to port 6666 and write to irc.erols.com.

After you've identified a hacker is on your system, and they appear to be using IRC, consider setting up a network sniffer. (Please make sure you talk to your institutions legal department and are aware of all applicable laws.) Network dumps are valuable because little, if any, IRC activity is encrypted. Even if a hacker uses an encrypted client to log in, such as SSH, the

actual connection to the IRC server will most likely be in clear-text.

TCPDump (available from ftp://ftp.ee.lbl.gov) is the standard packet sniffer on most Unix hosts. By default it only captures the first few bytes of every transaction: just enough to diagnose routing and network problems. When your interested in logging entire sessions it's important to read all available packet information. With TCPDump, the -s option controls how much data in each packet is collected. Consult your network MTU to determine the optimum setting. We use:

```
# /usr/sbin/tcpdump -n -s 1600 -F  -w tcpdump.
```

A quick and easy way to view the dump is to use the Unix command strings(1). If to much information is picked up, you can separate your tcpdump file using:

```
# /usr/sbin/tcpdump -r tcpdump. -w
```

and then run strings again on the output file. For example, if you're only concerned with IRC traffic (which normally is on port 6667), use:

```
# /usr/sbin/tcpdump -r tcpdump. -w irc. dst port 6667
# strings irc.
```

After gathering as much information about the hacker as possible through a packet dump and information from the various IRC configuration files, compile an M.O. (Modus Operundi) file. The M.O. should contain information such as the hacker's preferred nickname and any variations used, any dial-ups used, any related incidents, and any personal information discovered.

On several occasions I have picked up the exact age, name, and location of the hacker! This type data is invaluable when contacting law enforcement and correlating various incidents. I've found plotting the information on a map is a good way to provide a quick reference of active hackers.

On a slow afternoon I have also been known to go back to the M.O. files and check to see who is on IRC. If I believe I see the same hacker I'll send a quick note to the administrator of the domain, alerting them to a potential problem. Sometimes it turns out to be nothing, but the message is always appreciated.

I use ircii (available from http://www.irchelp.org/irchelp/ircii/), the classic UNIX irc client,

and primarily connect to EFnet. (Macintosh and PC users should check out http://www.irchelp.org for a list of clients). Generally the unix clients are safe as long as you use common sense. Don't accept files from strangers, don't run untrusted IRC scripts, and never run commands you don't understand. With most clients all IRC commands start with a forward slash ("/"). Everything else is a message sent to the channel.

After connecting, the first thing I do is start a log. With ircii, the command is:

```
/set log on
```

The logfile will be named IrcLog. To change names, type:

```
/set logfile
```

To look for a person, use the "who" command:

```
/who -nick  (looks for a particular nickname)
/who -host  (looks for anyone using a particular host)
```

Wildcards are allowed. However users marked as "invisible" will only show if you specify their exact nickname.

When checking IRC, be sure to look for all variations of the nickname. Hackers have the habit of logging in from a hacked site on a secondary nickname, while logged in with their primary nickname on their dial-in account. For example, perhaps there is a hacker who goes by the nickname "eleet". Querying IRC for eleet and eleet_ might show:

```
*       eleet   H*  user@ppp-7.isp.net
*       eleet_  H*  root@www.companyname.com
```

Chances are that www.companyname.com has been hacked. Even more interesting is that the person who did it probably, though not certainly as ident responses can be faked, is also using the dialup ppp-7.isp.net. There are two commands that can give you information: who and whois

```
/who -nick
```

will give you information on the nickname.

```
/whois
```

will give you a little more information.

Due to the number of hackers using IRC it is often the target of criticism, but there are thousands of legitimate users who use the IRC networks daily. Like any other community there will always be a criminal element. When hackers do use IRC it allows the administrator to monitor the criminal element and gain insight into their methodologies and habits. The acquisition of this knowledge can help system administrators, law enforcement, and security professionals track and prosecute hackers more effectively.

Reference sites:

- http://www.eggxpress.com - Information on eggdrop, BNC, and BitchX configuration files
- http://www.eggdrop.net - The home of eggdrop
- http://www.irchelp.org - Provides introductory documents and tutorials for using IRC
- http://metalab.unc.edu/dbarberi/papers/chats - Papers about the social perspective of IRC
- http://www.efnet.net - The popular EFNet, a dynamic IRC network and a favorite of many hackers
- http://www.undernet.org - The Undernet IRC network.
- http://www.newnet.net - The NewNET IRC network.
- http://www.self-evident.org - Dedicated to news on EFnet, including information on hackers and channels they frequent.
