Anomaly-based Botnet Server Detection

James R. Binkley
Computer Science Dept.
Portland State University
Portland, OR, USA
jrb@cs.pdx.edu

June 25, 2006

Abstract

We present simple statistical techniques for anomaly-based detection of on campus botnet servers. The techniques are derived from two experimental flow tuples that collect statistics based on four types of Layer 7 IRC messages including PRIVMSG, JOIN, PING, and PONG. These messages are used to extract a set of IRC channel names, the IP hosts in the channels, and message count statistics related to channel and host IRC usage. In the last year we have captured three instances of bot servers. In each case when compared to normal campus IRC servers, the bot server had certain highly anomalous statistics when compared to normal campus IRC servers.

1 Introduction

Botnets [1][4] are a modern scourge of the Internet and a cause of spam, denial of service attacks, and infected wormy hosts. Ourmon [2] is an near real-time network monitoring and anomaly detection tool. Originally our tool was intended as an open-source network monitoring system. Over time we discovered that it was also an anomaly-based detection system. We noticed that normal graphics or statistics were sometimes replaced by highly anomalous activity often associated with scanners, botnets, or DOS attacks. In the last year we have modified ourmon to detect botnets as well.

In this paper we look at two experimental IRC flow tuples that were created to primarily track IRC (and botnet client) networks. We then look at one example taken from Thanksgiving 2005 of how simple statistics associated with those IRC tuples proved to also catch a botnet server on our campus. In the last year we have captured multiple instances of botnet clients and three examples of botnet servers. For botnet client meshes, we have enough samples to be confident that our system works. (Please see our paper [3] for more information on botnet client mesh capture). For the botnet servers it is fair to say that this is not a large statistical sample, but on the other hand the statistics for capturing the botnet servers were very simple and may prove useful elsewhere. We therefore point out a number of anomaly-based simple statistical measures that have proven to be good botnet server indicators so far.

2 IRC tuple architecture

In our network-based collection system we collect two IRC tuples in thirty-second scanshots and then correlate the lists every hour over the day. This is because even in a large campus network such as ours with approximately 60000 packets per second peak traffic to the Internet, IRC data emerges in general very slowly. We assume we can capture up to 256 bytes worth of every packet including Layer-7 information such as IRC messages. IRC messages are parsed by a hand-coded C parser to form two tuples called the *node list* and *channel list*. The first list captures per host IRC information and the second organizes the hosts by IRC channel names (chatrooms). We are only interested in four kinds of IRC messages including PING, and PONG which are used to establish client server connectivity, and JOIN and PRIVMSG, both of which contain channel names. We do not store PRIVMSG IRC data.

Roughly the *node list* has the following structure and includes an IP address, as well as a number of statistical counters, and a list of channel names associated with the host.

```
(IPSRC, TOTALMSG, JOINS, PINGS, PONGS,
PRIVMSGS, CHANNELS, SERVERHITS}
```

TOTALMSG, JOINS, PINGS, PONGS, PRIVMSGS are counters. CHANNELS is the channel name list, and SERVERHITS tells us if the host is in anyway an IRC server.

The channel list has the following structure and includes the channel name, various statistics, and a list of IP hosts associated with the channel during the sample period.

```
(CHANNAME, HITS, JOINS, PRIVMSGS,
NOIPS, IP_LIST}
```

HITS, JOINS, PRIVMSGS are counters. NOIPS is the count of hosts in the channel, and the IP_LIST is the list of host IP addresses associated with the channel.

One additional IRC statistic gathered consists of total network counts of the four kinds of IRC messages seen by the collection system during a sample period.

3 Botserver Mesh Detection

Next we turn to botserver mesh detection using these statistics. We can distinguish at least four kinds of anomalies and there are probably more simply due to the large amounts of IRC traffic when compared to normal campus IRC traffic. Over Thanksgiving 2005, our campus experienced the largest botserver we have ever seen. We believe that the botserver in question at its peak had around 60000 off campus hosts reporting to it. We should point out that as is generally the case with anomaly detection, the analyst should have some notion of normal in terms of local domain statistics. In this case one has to have observed local IRC activity to know what is unusual. Even so, a bot server is easy to spot.

Outstanding anomalies included:

1. IP hosts per channel is a strong indicator of a botserver.
2. the number of messages for any given IRC server is a strong indicator.
3. lastly it can sometimes be the case that the overall number of scanning hosts counted in our entire network (and within the bot server's IRC channel) becomes very high.
4. PING and PONG message counts for our network domain as a whole are a strong indicator.

In terms of data presented by our system we note anomalies one, two, and three in our list above by looking at our "channels sorted by max messages" report subsection as seen in Table 1.

From experience we know that channel "bark" is benign (although somewhat automated in terms of both JOINS and PRIVMSGS), and in any case has a historically normal count of messages and hosts. On the other hand, channels f, x, and f-exp are new. The message count for f alone is historically unprecedented and an order of magnitude higher when compared to "bark". The above statistics are for the entire day, and busy chat channels on our campus driven by human text messages usually have no more than 2000 messages in one day. Therefore channel x and its 40965 PRIVMSGS are significant. Another significant anomaly appears with the ipcount (hosts per channel) for the three channels in question. On our campus we have never seen a normal chat IRC channel with more than 100 hosts in it. 20 is normal. Of course, further analysis shows that these numbers are even more significant because the three channels in question are shown elsewhere to share the same

Table 1: Malign IRC Botnet Server - Max Messages in Channels Report

channel	msgs	joins	privmsgs	ipcount	scanners
f	157403	156956	447	36727	1704
x	81161	40196	40965	13821	712
f-exp	20845	0	20845	5074	562
bark	11560	5509	6051	12	0

off campus botnet server. For reasons of space we cannot discuss the scanner count measure at length. Roughly it is a small statistical measure that shows how many hosts are "scanning". The scanners column in the table shows the count of alleged scanning IPs in the channel. In addition Figure 2 shows an anomalous amount of external hosts "scanning" our network. We discovered that these hosts were actually new bot clients attempting to connect over and over again to the bot server. Apparently the server had run out of TCP resources, thus new clients were not accepted.

Finally regarding anomaly number four – in a previous section we mentioned that we keep global count statistics for all IRC messages on our network and graph them using the RRDTOOL graph mechanism. Please refer to Figure 1. This graph shows a weekly view of counts of the four basic IRC message types. Normal daily counts for our message types are typically in the hundreds. Here we see that on Friday and Saturday PING and PONG messages suddenly are in the thousands. In all three bot servers we have seen on campus, PING and PONG messages were always elevated. PRIVMSG may or may not be elevated. All bot servers so far have significantly perturbed this simple graph.

4 Conclusion

Known techniques for botnet detection include honeynets and IDS systems such as snort[6] with signature detection. Honeynets [5] or darknets can certainly prove beneficial in terms of providing information about botnet technology.

We have recently published related work [3] on botnet-client detection using anomalies and present our botnet-server anomaly detection work here. We believe our anomaly-based work is an advance in the art, but ideally in the future we would like to be able to detect botnets via only layer 3 or layer 4 statistics as it is possible that layer 7 data may at some point be encrypted.

References

[1] P. Barford, V. Yegneswaran, An Inside Look at Botnets, *Special Workshop on Malware Detection, Advances in Information Security*, Springer Verlag, 2006

[2] J. Binkley, B. Massey, Ourmon and Network Monitoring Performance. *Proceedings of the Spring 2005 USENIX Conference, Freenix track*, Anaheim, April 2005.

[3] James R. Binkley, and Suresh Singh, An Algorithm for Anomaly-based Botnet Detection, *In Proceedings of Usenix Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI '06)*, San Jose, CA, July 2006.

[4] E. Cooke, F. Jahanian, and D. McPherson, The zombie roundup: Understanding, detecting and disrupting botnets. *In Proceedings of Usenix Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI '05)*, Cambridge, MA, July 2005.

[5] The Honeynet Project and Research Alliance. Know Your Enemy, Tracking Botnets. `http://honeynet.org/papers/bots`, March 2005.

[6] Snort IDS web page. `http://www.snort.org`, March 2006.

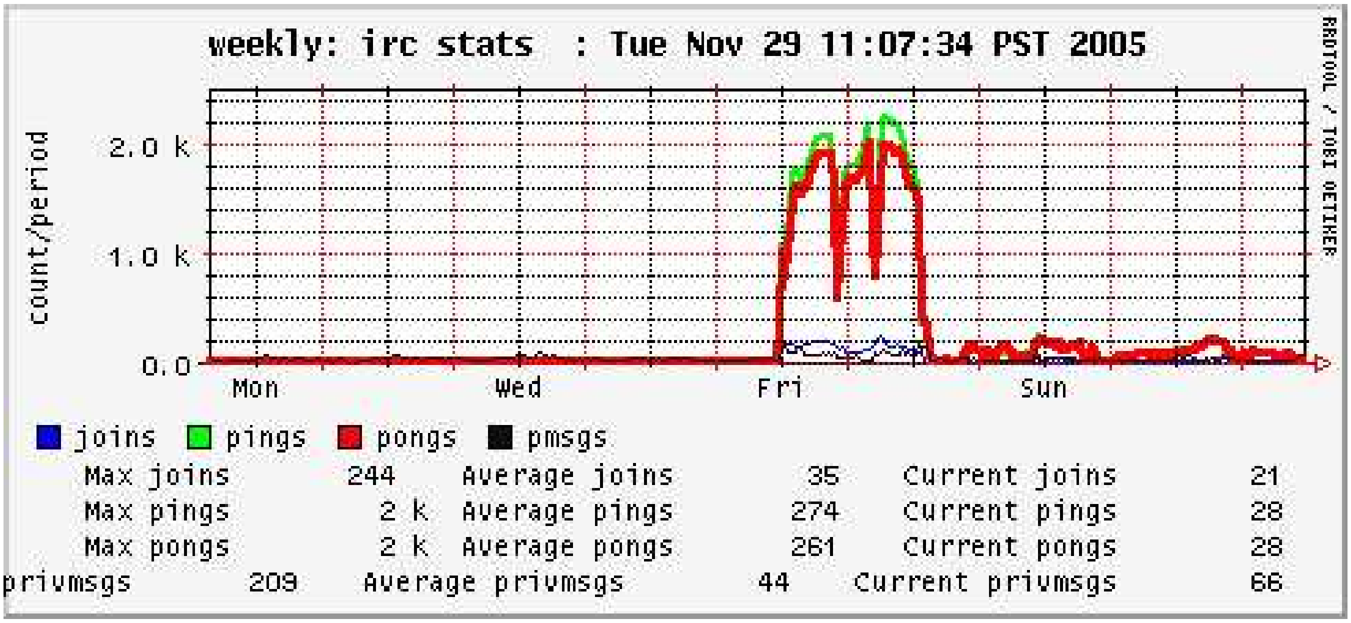

Figure 1: Thanksgiving Botserver IRC Message Counts

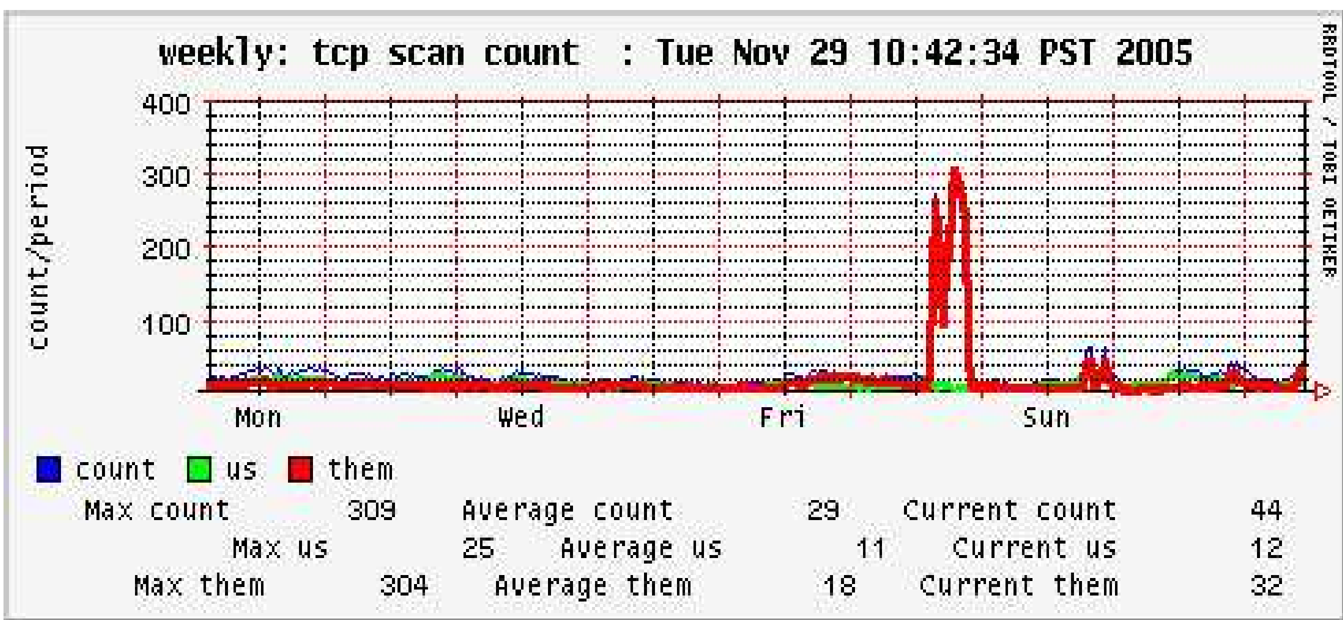

Figure 2: TCP Scanners During Bot Attack