2010 Ninth IEEE International Symposium on Network Computing and Applications

Quantification of Attackers Activities on Servers running Web 2.0 Applications

Katerina Goseva-Popstojanova, Risto Pantev, Ana Dimitrijevikj, and Brandon Miller
Lane Department of Computer Science and Electrical Engineering
West Virginia University, Morgantown, WV, 26506-6109, USA

***Abstract*—The widespread use of Web applications, in conjunction with large number of vulnerabilities, makes them very attractive targets for malicious attackers. The increasing popularity of Web 2.0 applications, such as blogs, wikis, and social sites, makes Web servers even more attractive targets. In this paper we present empirical analysis of attackers activities based on data collected by two high-interaction honeypots which have typical three-tier architectures and include Web 2.0 applications. The contributions of our work include in-depth characterization of different types of malicious activities aimed at Web servers that deploy blog and wiki applications, as well as formal inferential statistical analysis of the malicious Web sessions.**

I. Introduction

Many business and everyday activities nowadays depend on Web based applications. These applications face unique set of vulnerabilities due to the access by browsers, high exposure, and their integration with databases. SANS Institute Annual update of the top 20 security risks (http://www.sans.org/top20/) stated that almost half of the vulnerabilities discovered in 2007 were Web application vulnerabilities. Many organizations start to embrace Web 2.0 technologies to facilitate collaboration and to build new communication channels with customers, partners, and employees. Web 2.0 aims to enhance information sharing, collaboration, and functionality of the Web through social networking, video sharing, blogs, Web publishing, and other methods of information and content creation, editing, sharing, and distribution. However, these new technologies provide attackers with a broad range of new vulnerabilities and due to their interactive nature introduce higher security risk than traditional Web applications. Spammers and malware authors increasingly use Web 2.0 applications to carry out various attacks.

Finding attack attempts in a huge amount of monitored data from a Web server under regular use is a 'needle in a haystack' problem. Therefore, we decided to develop and deploy high-interaction honeypots that appear to be legitimate servers, but are actually collecting information on attackers' activities. In case of some honeypots the goal was to allow adversaries to easily penetrate the system, so researchers could study attackers' behaviors after successful exploitation [2], [11]. Our goal is different – we aim at studying the patterns and characteristics of attackers activities on typical servers that run Web 2.0 applications. Therefore, we deployed high-interaction honeypots with standard off-the-shelf operating system and applications that follow typical security guidelines and do not include user accounts with nil or weak passwords. Furthermore, instead of a set of independent applications typical for the honeypots in the related work, our honeypots have meaningful functionality and follow a three-tier architecture consisting of Web server, application server, and a database. Two widely used Web 2.0 applications - a blog and a wiki - were included in the honeypots' configurations. The main contributions of our work are as follows:

- Our analysis includes two sister case studies based on data collected by an advertised honeypot (using transparent linking) and an identical unadvertised honeypot. The unadvertised honeypot was used as control and allowed us to distinguish between attackers' activities based on search-based and IP-based strategies and quantify their contributions to the malicious Web traffic. Although specific types of malicious code that spread using popular search engines have been studied in the past (e.g., [15]), identification of different classes of malicious activities' that use search-based strategies and quantification of their contribution to the malicious traffic appear to be unique to our work.
- We provide an in-depth empirical analysis of attackers' activities classified as different types of vulnerability scans and attacks. A unique characteristics of our work is that we analyzed the HTTP application layer traffic at both request and session level, with specific focus on malicious sessions aimed at Web 2.0 applications which has not been done in the related work. This way we contribute towards establishing an evolving body of scientific knowledge about scans and attacks related to new technologies such as Web 2.0.
- We carried out inferential statistical analysis of the attackers activities, including several attributes of malicious Web sessions. Unlike the statistical characterization of the network traffic which has a long tradition (see for example [7], [9] and references therein), only very few attempts were made to statistically model some aspects of malicious traffic, such as [10], [4], and [8]. Neither of these studies included Web 2.0 applications and analysis of malicious Web sessions.

The paper is organized as follows. Section II presents

978-0-7695-4118-1/10 $26.00 © 2010 IEEE
DOI 10.1109/NCA.2010.21

the related work. The description of the experimental set-up and the approach used for classification of the malicious traffic are given in sections III and IV, respectively. In-depth analysis of the malicious Web traffic is presented in section V, while the inferential statistical analysis is given in section VI. Section VII concludes the paper.

II. Related work

Leurre.com (http://www.leurrecom.org/) is an example of data collection environment which is based on low-interaction honeypots that emulate particular operating systems and services. Analysis of frequently targeted ports, port sequences, and attack origins, based on data collected by multiple low-interaction honeypots was presented in [5], [14]. The analysis presented in [10] included using linear regression to model the number of attacks per unit of time as a function of attacks originating from a single country, and fitting a mixture of exponential and Pareto distributions to model the time between two consecutive attacks. One limitation of low-interaction honeypots is that attackers can only perform limited activities, without being able to scan for vulnerabilities or succeed in compromising the server.

In order to provide more realistic experience to the attackers and gather more information about attacks, high-interaction honeypots supported by the Honeynet Project (http://www.honeynet.org/) utilize actual operating systems and applications. The work presented in [12] was based on two low-interaction honeypots and one high-interaction honeypot. The analysis consisted of distribution of attacks across different ports, attacks origins, and description of two instances of successful attacks. Similar analysis based on three high interaction honeypots, each running different operating system, was presented in [6]. [13] explored whether port scans are precursors to attacks based on network traffic data collected from two high-interaction honeypots. The analysis based on the number of packets per connection distinguished among port scans, vulnerability scans and attacks. However, the distribution of scans and attacks across different applications was not addressed in [13] since it had a different goal. The work presented in [2] analyzed the behavior of the attackers who succeeded in breaking into a high-interaction honeypot which had weak passwords for multiple SSH user accounts.

A recent work presented in [4] compared the data collected by Leurre.com and two high-interaction honeypots which ran several unrelated applications. The analysis was based on the traffic at network layer and included most often scanned ports, number of attacking hosts, persistence of attackers, and the distribution of the time between the first packet exchanges from reappearing IPs. Another recent paper [3] compared the events that targeted similar ports on the same day across data collected by two high-interaction honeypots and data from two global repositories.

Our previous work [8] was based on honeypots running on Linux operating system, with a three-tier architecture consisting of Apache Web server, PHP5 server with phpMyAdmin application used as a front-end of an MySQL database. In this paper, we also follow a three-tier architecture, this time using honeypots running Windows XP operating system, with Microsoft IIS Web server, PHP server and MySQL database and include two Web 2.0 applications: a Wiki and a Blog. Consequently, the observed attackers' activities were very different, with significantly more HTTP traffic reaching the Web 2.0 honeypots compared to non-Web 2.0 honeypots in [8]. Even more, 80% of HTTP sessions on the advertised Web 2.0 honeypot included one or more Web 2.0 components, thus allowing us to observe very different categories of vulnerability scans and attacks. Finally, while the statistical analysis in [8] was focused on the TCP traffic, in this paper we analyze the malicious Web sessions at the application layer, with specific focus on vulnerability scans and attacks aimed at Web 2.0 applications. None of the related works have focused on these aspects of the malicious traffic.

III. Experimental setup

Honeypots used in our work follow the general principles of generation II high-interaction honeypots developed as a part of the Honeynet project. Our experimental setup, including the configuration deployed on each honeypot, is presented in Figure 1. An integral part of the setup is the honeywall which acts as a bridging firewall between the honeypot and the Internet. The honeywall logs all of the packets using TCPDump and then forwards the traffic to the honeypots without modifying the hop count of the packets. The honeywall also limits the outbound connections an attacker can initiate from a honeypot, which reduces the risk of malicious activities originated from a compromised honeypot. The captured network traffic was stored in a central data repository which ran on a separate physical host. In addition, we collected information related to the system activity and various applications running on our honeypots. We also instrumented each Web page (static or dynamic) on both honeypots to use Google Analytics for collecting and logging data on visits by human users.

We built two identical honeypots. Each honeypot had its own IP address and a hostname and ran on a VMWare virtual machine on an Ubuntu 8.10 Server (kernel: 2.6.27-11-server) host operating system. One of the honeypots was *advertised* using a technique called 'transparent linking' which involves placing hyperlinks pointing to the honeypot on a regular, public Web page, so that the advertised honeypot is indexed by search engines and Web crawlers, but cannot be accessed directly by humans. This way we allow for attacks based on search engines (using the so called *search-based strategy* [11]). The second honeypot was not advertised anywhere on the Web. This *unadvertised* honeypot could only be reached by *IP-based strategy* when an attacker scans an

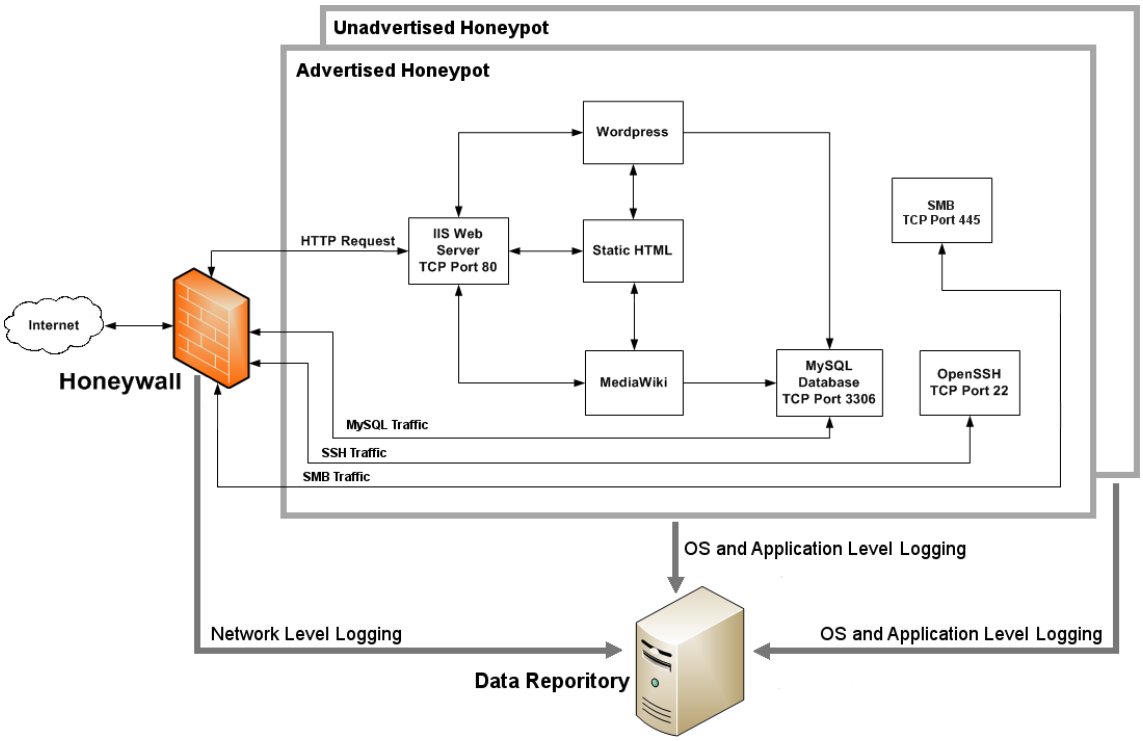

Figure 1. Experimental Setup

IP address [11]. In our setup the unadvertised honeypot served as a control and allowed us to determine the relative contribution of search-based strategies (which only work on the advertised honeypot) to IP-based strategies (which work on both honeypots).

The operating system of each honeypot is Windows XP Service Pack 2. Each of our honeypots ran a Web based system with a three-tier architecture (i.e., Web server, an application server, and database). The configuration consisted of Microsoft's Internet Information Services (IIS) Web server version 5.1, PHP Server version 5.0.2, and MySQL Server version 4.1. In addition, two open source Web 2.0 applications were installed on each honeypot: the wiki software MediaWiki (version 1.9.0) which is used as the application base for Wikipedia and the blogging software Wordpress (version 2.1.1). According to [16], MediaWiki is the most widely used wiki software and Wordpress is the most downloaded open source content management system. We generated content for each Web 2.0 application, so it would appear that they were being actively used. The MySQL server contained one database for each of the Web 2.0 application, as well as a system database.

All software packages installed on the honeypots are typical installations of somewhat older versions, each with a number of known vulnerabilities. Such configurations provided plenty of opportunities for compromising the honeypots, while still running applications new enough to be found on Internet. All user accounts at our honeypots (two in Windows XP, nine in Wordpress, five in MediaWiki, and seven in MySQL on each honeypot) had strong passwords in order to prevent simple password cracking attempts from succeeding. It should be noted that each Web 2.0 application was configured to accept anonymous submissions.

IV. Classification of Attackers' Activities

Our honeypots ran during a period of almost four months (March 30 to July 26, 2009). We used custom developed scripts to parse the network traffic capture file and application level logs. For both honeypots we first removed the legitimate non-malicious traffic which consisted of the system management traffic and legitimate Web crawlers such as Google and MSNbot. The crawlers were removed based on the IP addresses listed in iplists.com and other similar sites and based on manual inspection of the remaining traffic. We analyzed only the incoming traffic because the outgoing traffic consisted only of responses to requests sent to the honeypots. It should be noted that neither of our honeypots was exploited successfully in the four months duration of the experiment.

As expected the traffic was dominated by the TCP component. Thus, 99.55% of packets on the advertised and 98.75% of packets on the unadvertised honeypot were due to TCP traffic. Since TCP is connection oriented protocol, we define a *connection* as a unique tuple {source IP address, source port, destination IP address, destination port} with a maximum inter-arrival time between packets of 64 seconds

Table I. BASIC STATISTICS ABOUT TCP PORTS VISITED ON EACH HONEYPOT

Port	Advertised honeypot				Unadvertised honeypot			
	Connections		Packets		Connections		Packets	
HTTP (80)	10,806	44.10%	133,998	52.80%	9,025	38.74%	56,154	31,75%
SSH (22)	9,154	37.36%	106,604	42.01%	8,522	36.58%	99,057	56.00%
SMB (445)	3,365	13.73%	11,199	4.41%	3,959	16.99%	18,445	10.43%
Other	1,177	4.80%	1,966	0.77%	1,792	7.69%	3,232	1.83%
Total	**24,502**	**100.00%**	**253,767**	**100.00%**	**23,298**	**100.00%**	**176,888**	**100.00%**

following the definition used in network traffic analysis [9].

We first address the distribution of the TCP traffic across destination ports. The following observations can be made based on results presented in Table I. HTTP (port 80) traffic was significant on both honeypots contributing to 44.10% of the connections on the advertised and 38.74% on the unadvertised honeypot. This is a significant increase compared to HTTP contribution on the non-Web 2.0 honeypots in our earlier work [8] which was slightly over 1% on advertised and less than 1% on unadvertised honeypot. SSH (port 22) was the second most popular port, with almost the same percentage of connections on the advertised and unadvertised honeypots. Our further analysis showed that over 99% of the SSH packets on each honeypot were part of dictionary password cracking attacks, which indicates that using weak passwords may still be among the weakest links in systems security. The third most popular port was 445, which is used by SMB (Server Message Block) protocol for file sharing in Windows operating systems. Interestingly, MySQL database server received very little (i.e., less than 0.1%) traffic directly on port 3306 on any honeypot.

The rest of the paper is focused on analysis of the malicious HTTP traffic at the application layer, which showed the richest set of attackers activities including those aimed at Web 2.0 applications. One of the main contributions of this paper is an in-depth analysis of attacker activities classified as port scans, vulnerability scans, and attacks. In this context, a *port scan* is used to check for open ports and active services. A *vulnerability scan* is used to explore the presence of a vulnerability. Finally, an *attack* is defined as an exploit of vulnerabilities by a human or a malicious program.

HTTP applications logs, which are rich sources of information related to user activities, appeared to be very useful in the process of mapping the malicious traffic to different classes. For both vulnerability scans and attacks coming through the front-end IIS server, we specifically distinguished between those ending up at IIS server and those spanning multiple components of the Web-based system, which is unique to our study. Furthermore, in addition to the request level traffic, we analyzed the session level traffic, where a *session* is defined as a sequence of requests from the same source IP address to port 80, with a time between two successive request not exceeding 30 minutes [7].

The approach used for labeling the malicious traffic is very important for the accuracy of the analysis. However, labeling malicious traffic is not trivial and with the current state-of-the-art in the area of intrusion detection cannot be done automatically. Some of the related work, such as [13], used heuristic based on the number of packets to classify the network traffic as port scans, vulnerability scans, and attacks. Instead of using heuristics, we decided to use a semiautomated process based on identification of patterns in the HTTP application level logs. We chose this approach not only because it improves the accuracy of the mapping process, but also because it allows us to identify different classes of vulnerability scans and attacks.

Since the textual format of the log data is not suited for flexible, customized analysis, we used our custom developed tool to parse the IIS logs and include the log entries in a relational database. Out of the total 9,868 HTTP requests we identified 1,610 unique requests. Classification of these 1,610 requests was based on the specific patterns of attackers' activities. For this we looked at different fields of the HTTP requests, such as the method used, values passed to the parameters, agent field, bytes transferred, error code, etc. For example, any request that posted spam to the wiki had to use `POST` method and pass `submit` value to the `action` parameter (i.e., `action=submit`) in a request to the `/wiki/index.php` page. Once the pattern that represents a specific activity was identified, we queried the database and labeled the corresponding requests. This process was repeated for each recognizable attackers' activity. We also searched the publicly available vulnerability databases such as http://web.nvd.nist.gov or http://www.securityfocus.com/ for a specific signatures seen in the requests on our honeypots. For example, it was found that the string `autoLoadConfig[999][0][loadFile]=http://` corresponds to the know attack CVE-2006-4215.

After labeling all requests, we examined each session and based on the labels of the requests in that session classified it in one of the classes given in Table II. (More details about these classes are presented in the next section.)

Once labeling HTTP requests and sessions was completed, labeling the TCP traffic was fairly simple and done automatically by comparing the IP addresses and time stamps of the TCP connections/packets and the HTTP sessions/requests. TCP connections to port 80 that were not part of an HTTP session were labeled as port scans using the characteristics of the TCP protocol, similarly as in [13].

V. Description of attackers' activities

The analysis presented in this section consists of *descriptive statistics* of the attackers' activities classified to port scans, vulnerability scans, and attacks on different components of the Web-based system.

Port scans contributed only to 1.93% of connections to port 80 on the advertised honeypot and 0.68% on the unadvertised honeypot. Ten attackers on advertised honeypot and five on unadvertised honeypot (with four being common) first scanned port 80 before attacking the Web server.

The breakdown of the Web traffic on vulnerability scans and attacks shown in Table II allows us to make the following observations:

- Close to 74% of all HTTP sessions on the advertised honeypot (i.e., 87% on the unadvertised honeypot) were due to vulnerability scans, compared to 26% attacks (i.e., 13% on the unadvertised honeypot). However, the number of requests included in attack sessions was higher (close to 56% on advertised and 85% on unadvertised homeypot), mainly due to a DoS attacks aimed at IIS which had sessions with many requests.
- Over 80% of sessions (i.e., 55% due to vulnerability scans and 25% due to attacks) on the advertised honeypot were aimed at least at one Web 2.0 component. With an exception of only several sessions, no such malicious activities were noticed on the unadvertised honeypot, which shows that attackers used search-based strategy to reach the Web 2.0 components on the server. This distinct behavior, which is obvious from Figures 2 and 3, clearly illustrates that using unadvertised honeypots, as in most of the related work, would have shown only a very small piece of attackers' activities aimed at Web 2.0 applications.

Next we present more detailed analysis of attackers' activities, including the characteristics of the malicious HTTP sessions in terms of the following attributes: Number of requests per session, Session duration, and Bytes transferred per session. The box plots of these attributes for different types of attackers' activities[1] are shown in Figure 4. We start with describing different types of *vulnerability scans*.

DFind allows an attacker to probe whether a Web server (in our case IIS) is vulnerable to specific exploits. Attackers used IP-based strategy to reach both servers. (In the Venn diagrams DFind is included in the 'Static$^+$' category since it was not intended towards Web 2.0 content.)

Static$^+$ category includes malicious sessions which browsed static content (html pages, pictures, and video files). It also includes sessions in which attackers were searching for non-existing content and applications on the honeypots, which constituted 1.70% of the sessions on the advertised honeypot and 22.08% of the sessions on the unadvertised

[1]Several less interesting activities, each with small number of sessions are not shown in Figure 4 due to lack of space.

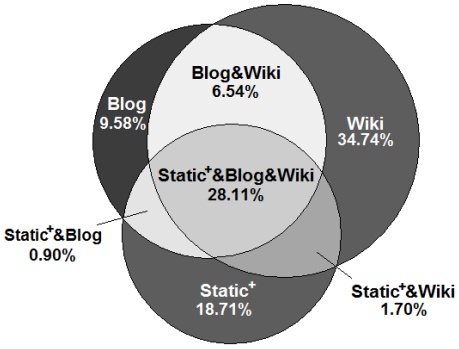

Figure 2. HTTP sessions on advertised honeypot

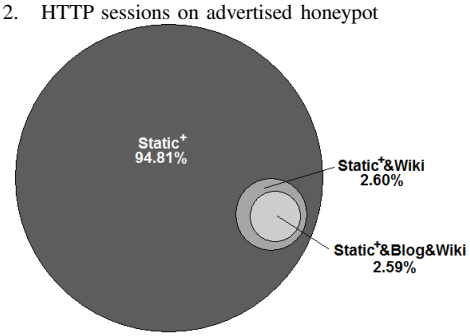

Figure 3. HTTP sessions on unadvertised honeypot

honeypot. As it can be seen from Figure 4, Static$^+$ sessions had considerable variation in all three session attributes and not surprisingly some of them had the highest number of bytes transferred among all malicious sessions.

At least one Web 2.0 application was fingerprinted in majority (i.e., 75%) of vulnerability scan sessions on the advertised honeypot, which constituted over 55% of the total number of malicious sessions. In most cases attackers accessed the Blog and Wiki applications either directly through the specific URLs that led to these two applications or through the homepage by following the links leading directly to these applications. These vulnerability scans exclusively related to Web 2.0 applications (i.e., Blog, Wiki, or combination of both) consisted close to 51% of the total number of malicious sessions. Another interesting observation is that the Wiki was targeted close to four times more often than the Blog. Most Blog and Wiki sessions and combination of both had only one request and small number of bytes transferred. Even the sessions with the longest durations did not have many requests. Close to 39% of unique IP addresses that scanned Blog and Wiki applications originated from the USA, followed by 18% from Canada and 17% from China. Blog and Wiki applications were fingerprinted in conjunction with Static$^+$ content only in around 5% of sessions, which further confirms that attackers' attention is drawn directly towards the Web 2.0 applications. The fact that only three sessions on the unadvertised honeypot included vulnerability scans of Web 2.0 applications (compared to 619 sessions on the advertised honeypot) is a clear indication that attackers use search-based strategies to

Table II. BREAKDOWN OF VULNERABILITY SCANS AND ATTACKS

	Advertised honeypot				Unadvertised honeypot			
	Sessions		Requests		Session		Requests	
Vulnerability scans: Total	**824**	**73.77%**	**4,349**	**44.07%**	**67**	**87.01%**	**1,361**	**15.35%**
DFind	24	2.15%	25	0.25%	23	29.87%	24	0.27%
Static$^+$ (S$^+$)	181	16.20%	1,522	15.42%	41	53.25%	1,243	14.02%
Blog	107	9.58%	253	2.56%	0	0.00%	0	0.00%
Wiki	385	34.47%	923	9.35%	0	0.00%	0	0.00%
Blog & Wiki (B&W)	73	6.54%	406	4.11%	0	0.00%	0	0.00%
Static$^+$ & Blog	10	0.90%	72	0.73%	0	0.00%	0	0.00%
Static$^+$ & Wiki (S&W)	19	1.70%	319	3.23%	2	2.60%	65	0.73%
Static$^+$ & Blog & Wiki (S&B&W)	25	2.24%	829	8.40%	1	1.30%	29	0.33%
Attacks: Total	**293**	**26.23%**	**5,519**	**55.93%**	**10**	**12.99%**	**7,504**	**84.65%**
DoS	4	0.36%	3,724	37.74%	9	11.69%	7,490	84.49%
Spam on the Blog (SpamB)	23	2.06%	82	0.83%	0	0.00%	0	0.00%
Spam on the Wiki (SpamW)	249	22.29%	1,217	12.33%	0	0.00%	0	0.00%
Password cracking Blog user accounts (PassB)	9	0.81%	127	1.29%	0	0.00%	0	0.00%
RFI	4	0.36%	13	0.13%	0	0.00%	0	0.00%
SQL injection	1	0.09%	20	0.20%	0	0.00%	0	0.00%
RFI & SQL injection	1	0.09%	14	0.14%	1	1.30%	14	0.16%
XSS	2	0.18%	322	3.26%	0	0.00%	0	0.00%
Total	**1,117**	**100%**	**9,868**	**100%**	**77**	**100%**	**8,865**	**100%**

locate servers that host Web 2.0 applications.

Next, we describe the Web sessions classified as *attacks*. *Denial of Service (DoS) attack* was based on the Microsoft IIS WebDAV PROPFIND and SEARCH Method Denial of Service Vulnerability. The attack was unsuccessful because this vulnerability was fixed in Windows XP SP2 which ran on our honeypots, thus resulting in requests with 404 errors. This attacks originated from the same IP address and likely used IP-based strategy to reach both servers. Although the number of sessions on each server was small, they contributed to significant number of malicious requests (i.e., over 3,700 requests on advertised server and close to 7,500 requests on unadvertised server). This distinct behavior of DoS attacks is easily observable in Figure 4.

Posting *spam messages* was part of the majority of malicious attack sessions on the advertised honeypot (i.e., close to 93%). Spam obviously is becoming a major problem for many servers that host Web 2.0 applications due to their interactive nature. Spam on the Blog is in a form of comments to already posted discussion topics. In our case comments contained random text or advertisements and almost always link(s) to other Web site(s). Spam on the Wiki is in a form of posting a new topic or editing existing topics with random text or advertisements. Most of Wiki postings in our case were tailored to look like new topics, each including link(s) to other Web site(s). We tested the malicious nature of the links included in the Blog and Wiki with the Google's Safe Browsing diagnostic page and by manual inspection. It appeared that none of the links contained malicious content, that is, they all led to Web sites with spam like content. As in the case of vulnerability scans, significantly more spam sessions were aimed at the Wiki compared to the Blog (i.e., 249 versus 23 sessions). Furthermore, 79% of the 249 spam sessions on the Wiki included positing more than one spam message, compared to only one such session (out of 23) on the Blog. These phenomena are likely due to the fact that Wiki allows greater freedom of access and posting. Most of the spam sessions included small number of requests (with a median of 4 requests) limited to opening one page and posting spam message(s). As it can be seen in Figure 4, spam sessions on the Wiki had more requests and bytes transferred and were longer than spam sessions on the Blog. The session with most requests had 247 requests and besides posting spam included access to the Blog and Wiki through the homepage and traversal of topics on the Wiki page. Another interesting point is that close to 63% of unique IPs that posted spam on our Web 2.0 applications were from China, followed by over 9% of unique IPs located in the USA, despite the fact that the Blog and Wiki applications were scanned less often from IP addresses located in China than in the USA. The fact that no spam ended on the unadvertised server indicates the use of search-based strategies.

Only a few attack sessions were aimed at *password cracking* user accounts, specifically belonging to the Blog. The attackers that posted spam content on the Wiki created their own accounts and did not try to guess passwords.

The advertised server also experienced some amount of *Remote File Inclusion (RFI)*, *SQL injections* and *Cross-site Scripting (XSS)*. RFI attacks reached only the advertised honeypot. Two of them tried to exploit CVE-2006-4215 and CVE-2006-3771 and the remaining two could not be related to any known attack. The specific SQL injection attack on the advertised server was based on the CVE-2007-2821 vulnerability in WordPress before 2.2 which allows remote attackers to execute arbitrary SQL commands via the cookie parameter. An RFI attack in combination with an SQL injection attack was aimed towards non-existing pages on our honeypots. This is the only attack, in addition to the DoS attack, which ended on both honeypots and thus used

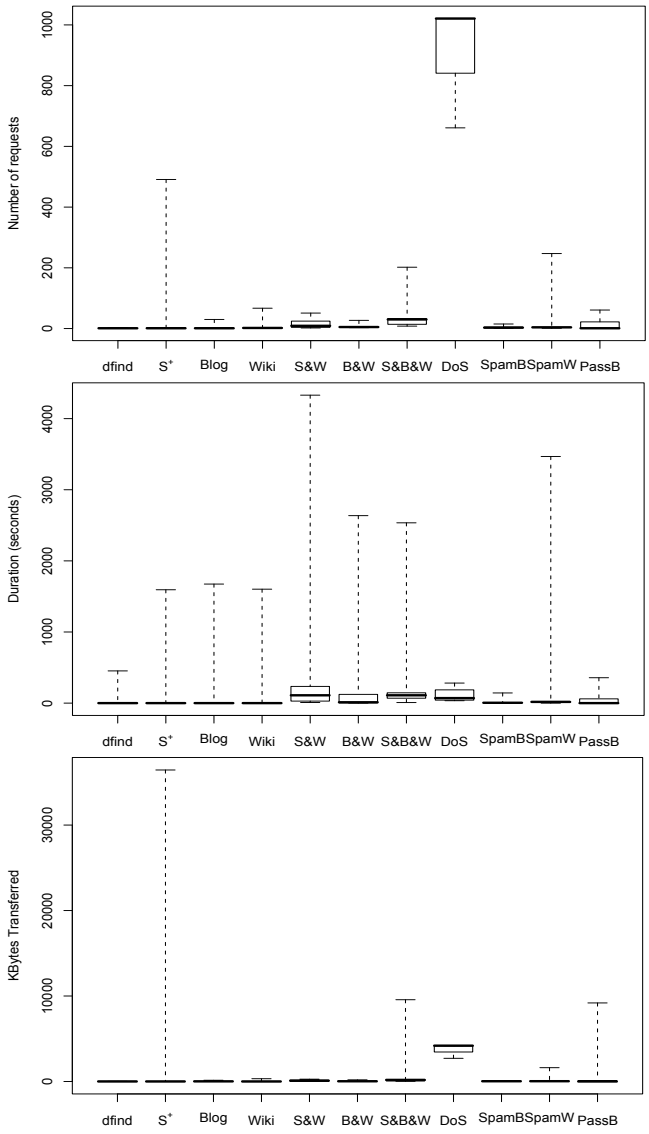

Figure 4. Box plots of HTTP session attributes

an IP-based strategy. Two attackers explored the CVE-2007-0308 XSS vulnerability toward non-existing functionality on the advertised honeypot.

After presenting the detailed description of the malicious HTTP traffic, we make several remarks. (1) At this point it is interesting to revisit the results presented in Table I and the fact that in this experiment we observed an increase in the malicious HTTP traffic compared to our previous experiments with non-Web 2.0 honeypots [8]. In the case of the advertised honeypot the HTTP traffic was mainly due to connections accessing Web 2.0 applications (around 21% of the total TCP connections) and to the Denial of Service attack on Microsoft IIS (around 14% of the total TCP connections). The HTTP traffic on the unadvertised honeypot showed different trend – it was mainly due to the Denial of Service attack which had twice as many connections on that honeypot (slightly over 32% of all TCP connections). Both servers also exhibited some increase in the access to the static Web content. (2) Our initial classification of vulnerability scans and attacks was based on a large number of fine-grained classes. Consolidating the initial fine-grained classes into smaller number of coarse-grained classes shown in Table II allowed us to present the main patterns of attackers' behavior in a limited space. (3) It should be noted that close to 19% of attack sessions to the advertised server were pure attacks; each of the remaining attack sessions was a combination of a vulnerability scan and the actual attack in a single session. (4) It appears that vast majority of the attackers' activities were based on running automatic tools that crawled Web scanning for vulnerabilities or launching attacks[2]. A preliminary analysis based on Google Analytics statistics, which only captures human-generated activity, indicated that only a very small amount (i.e., 3.5%) of the HTTP sessions were human-generated, and most, if not all, of those sessions were malicious. Accurate distinction between automatic and human-generated sessions is an open research problem and further analysis of this topic is out of the scope of this paper. (5) As expected, malicious crawlers typically did not visit the `robot.txt` file. Out of seventeen that did, three intentionally explored the pages they were not supposed to visit accordingly to `robot.txt`.

VI. Inferential statistical analysis

The statistical characterization of the malicious traffic in this section is focused only on the advertised honeypot which more realistically represents the actual attackers' activities. Before presenting the results of the distribution fitting to the session attributes Number of requests, Session duration, and Bytes transferred per session, we first explore the 3D scatter plot of the malicious Web sessions shown in Figure 5. We observed that Web 2.0 sessions can have long duration, but not as many requests and bytes transferred as non-Web 2.0 sessions. As it can be seen from Figure 4 relatively short sessions with large number of request are due to DoS attack, while the sessions with most bytes transferred are vulnerability scans that include the 'Static^{+}' category. It is interesting to note that there are no sessions with all three attributes having significantly large values (i.e., no points exists along the diagonal of the cube in Figure 5).

Based on the descriptive statistical analysis presented in section V and the 3D scatter plot of the malicious HTTP sessions we suspected that heavy-tailed distributions may be a good model for some characteristics of the malicious traffic. The simplest heavy-tailed distribution is the classical Pareto distribution which has the cumulative distribution function (CDF) $F(x) = P[X \leq x] = 1 - (b/x)^{\alpha}$. In practical terms, a random variable that follows a heavy-tailed

[2]Note that we have excluded from the dataset all known legitimate crawlers that retrieve content for indexing. Also note that human users cannot directly access the honeypot advertised with transparent linking.

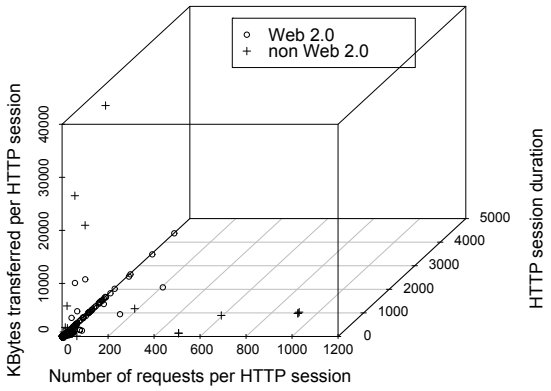

Figure 5. Malicious HTTP sessions

distribution can give rise to extremely large values with non-negligible probability. To estimate the tail index α of a Pareto distribution we employ the *log-log complementary distribution (LLCD) plots* and *Hill estimator* [7]. We used the Anderson-Darling (A^2) goodness-of-fit-test [1] because it is generally more powerful for detecting deviations in the tail of a distribution than the better known Kolmogorov-Smirnov and χ^2 tests. All test of hypothesis were done at significance level of 0.05.

Despite the fact that most of the malicious Web sessions had small number of requests and bytes transferred and short duration (i.e., they are close to the origin in Figure 5), sessions attributes appeared to have heavy tails. As it can be seen from Table III, the tails of the Number of requests, Session duration, and Bytes transferred all follow Pareto distribution with $\alpha < 1$, that is, have both infinite mean and variance. It is interesting to explore what are the sessions that ended up in the heavy-tails of each attribute.

Among the 100 sessions in the tail of the Number of requests 79% were vulnerability scans and 21% were attacks. The three sessions with most requests each had 1,021 requests and belonged to the DoS attack. These sessions, however, did not have long duration in time. Besides the DoS sessions, among the sessions in the tail of the Number of requests were vulnerability scans in which attackers accessed Static$^+$ content. Some of these sessions had as many as 491 requests, but zero bytes transferred since attackers were looking for applications such as PHPmyAdmin that were not deployed on our honeypots. Some of the sessions in which attackers posted spam on the Wiki were also in the tail of this attribute, as well as some vulnerability scans in which attackers accessed a combination of Static$^+$, Blog, and Wiki content. The later were among long sessions with many bytes transferred.

The tail of the Session duration attribute consisted of 300 points. As it can be seen in Figure 4 Session duration attribute has the highest variability across many different types of malicious Web sessions. Among these sessions 47% were labeled as vulnerability scans and 53% as attacks. The longest session lasted 72 minutes and belonged to a vulnerability scan in the category 'Static$^+$ & Wiki'. The second longest session, which lasted 58 minutes, was classified as 'Spam on the Wiki'. The longest sessions in time units, however, did not have many requests and bytes transferred. In the tail of the Session duration attribute, there were also other sessions from 'Spam on the Wiki' category, and vulnerability scans from the 'Blog & Wiki' and 'Static$^+$ & Blog & Wiki' categories. Even though most of the vulnerability scans to 'Blog' or 'Wiki' were short in duration, there were some sessions that were long and appeared in the tail of Session duration attribute.

Table III. DISTRIBUTIONS OF SESSION ATTRIBUTES

	# of requests	Duration	Bytes transferred
min/median/max	1/2/1021	0/2/4330 sec	0/9.2KB/35.6MB
Distribution	Pareto	Pareto	Pareto
Parameters	$\alpha = 0.8$ $b = 6$	$\alpha = 0.6$ $b = 14$	$\alpha = 0.6$ $b = 36035$

Out of the 100 sessions in the tail of the Bytes transferred attribute, 67% were vulnerability scans and 33% were attacks. The three sessions with most bytes transferred (i.e., 17MB, 23MB, and 35MB) were vulnerability scans labeled as 'Static$^+$'. In the tail of the Bytes transferred attribute were also some of the password cracking attacks on Blog user accounts, DoS sessions, and some vulnerability scans from the 'Static$^+$ & Blog & Wiki' category.

Further, we explore whether HTTP sessions and requests generated by unique source IP addresses can be modeled with heavy-tailed distributions. The number of sessions per unique IP was modeled well with a lognormal distribution with parameters $\sigma = 0.70538$ and $\mu = 2.0034$. Lognormal distribution is a skewed distribution, which unlike the heavy-tailed Pareto distribution has a finite variance. On the other side, the number of requests per unique attacker was modeled well with Pareto distribution with parameters $\alpha = 1.3$ and $b = 26$. In practical terms, a heavy-tailed distribution means that extremely large number of request can originate from a small number of attackers with non-negligible probability. In our case, for example, the DoS attack to IIS which originated from one IP address contributed to the almost 38% of requests on the advertised honeypot and over 84% of requests on the unadvertised honeypot.

VII. CONCLUSION

In this paper we presented an empirical analysis of attackers activities on multi-tier Web servers which include Web 2.0 applications. The analysis was based on data collected by two high-interaction honeypots, during a period of almost four months. An important observation is that over 80% of the sessions on the advertised honeypot involved at least one Web 2.0 application, which undoubtedly illustrates attackers' interest in exploiting these highly interactive Web technologies. Specifically, around 51% of the total malicious sessions

on the advertised honeypots scanned the Blog or Wiki or combination of both either directly accessing the URLs or following the link to these applications directly from the homepage without accessing other static content and/or other applications. We also noticed that around 24% of the malicious sessions included posting spam messages, which dominates the attacks sessions on the advertised honeypot. Common to vulnerability scans and attacks on Web 2.0 components is that the Wiki was targeted significantly more often than the Blog, likely due to its richer functionality and greater freedom of access and postings.

The fact that almost no malicious activity aimed at Web 2.0 applications was registered on the unadvertised honeypot clearly illustrates that attackers use search-based strategies to access these applications. It also illustrates that using unadvertised honeypots, as in most of the related work, would show very unrealistic view on attackers activities.

Based on our analysis it appears that when it comes to attacking 'any server' rather than major governmental or e-commerce servers, the easiest ways to attack the server seem to dominate. In our case, these included password cracking attacks on SSH and Blog user accounts and vulnerability scans and attacks (in a form of spam) on Web 2.0 applications which due to their interactive nature allow access by default. Significantly less malicious activity was aimed at specific known vulnerabilities.

The inferential statistical analysis of the malicious Web sessions showed that the Number of requests, Session duration, and Bytes transferred all follow Pareto distributions with infinite mean and variance. The percentage of Web 2.0 related sessions in the tails of each attribute approximately followed the overall contribution of around 80%. We also showed that the number of requests generated by unique attackers follow heavy-tailed distribution with a small number of attackers generating most of the malicious traffic.

As a concluding remark, the two main contributions of this paper are (1) in-depth analysis of malicious traffic aimed at Web servers that run widely used Web 2.0 applications and (2) inferential statistical analysis of the characteristics of the malicious HTTP traffic. Unlike research and practice of other quality attributes such as performance and reliability/availability, quantification and statistical characterization of malicious traffic and security in general are in their infancy. The presented results in this paper are a step towards filling this gap. Quantification and statistical models like these can be used for generating realistic malicious traffic for verification and validation of systems' security (similarly as benchmarks are used for performance testing and evaluation) or to help the intrusion detection process.

ACKNOWLEDGMENTS

This work is funded by the NSF CAREER grant CNS-0447715. The authors thank David Krovich and Jonathan Lynch for their support with the experimental setup.

REFERENCES

[1] T. W. Anderson and D.A. Darling, "A test of goodness of fit," *J. Amer. Stat. Assn.*, vol.49, pp. 765-769, 1954.

[2] E. Alata, V. Nicomette, M. Kaaniche, M. Dacier, and M. Herrb, "Lessons learned from the deployment of a high-interaction honeypot," *6th European Dependable Computing Conf.*, pp. 39-46, 2006.

[3] R. Berthier, D. Kormann, M. Cukier, M. Hiltunen, G. Vesdonder, and D. Sheleheda, "On the comparison of network attack datasets: An empirical analysis," *11th IEEE High Assurance Systems Eng. Symp.*, pp. 39-48, 2008.

[4] R. Bloomfield, I. Gashi, A. Povyakalo, and V. Stankovic, "Comparison of emirical data from two honeynets and a distributed honeypot network," *19th Int'l Symp. Software Reliability Engineering*, pp. 219-228, 2008.

[5] P. T. Chen, C. S. Laih, F. Pouget, and M. Dacier, "Comparative survey of local honeypot sensors to assist network forensics," *1st Int'l Workshop on Systematic Approaches to Digital Forensic Eng.*, pp. 120-132, 2005.

[6] M. Dacier, F. Pouget, and H. Debar, "Honeypots: Practical means to validate malicious fault assumptions," *10th IEEE Pacific Rim Int'l Symp. Dependable Comp.*, pp.383-388, 2004.

[7] K. Goseva-Popstojanova, F. Li, X. Wang, and A. Sangle, "A Contribution towards solving the Web workload puzzle," *36th IEEE/IFIP Int'l Conf. Dependable Systems & Networks*, pp. 505-514, 2006.

[8] K. Goseva-Popstojanova, B. Miller, R. Pantev, and A. Dimitrijevikj, "Empirical Analysis of Attackers Activity on Multi-Tier Web Systems." *23rd Int'l Conf. Advanced Information Networking and Applications*, pp. 781-788, 2010.

[9] N. Hohna, D. Veitch, and T. Ye, "Splitting and merging of packet traffic: Measurement and modelling," *Performance Evaluation*, vol.62, pp. 164-177, 2005.

[10] M. Kaaniche, E. Alata, V. Nicomette, Y. Deswarte, and M. Dacier, "Empirical analysis and statistical modelling of attack processes based on honeypots," *Workshop on Empirical Evaluation of Dependability and Security*, 2006.

[11] http://www.honeynet.org/papers/webapp/

[12] R. McGrew and R. B. Vaughn, "Experiences with honeypot systems: Development, deployment, and analysis," *39th Annual Hawaii Int'l Conf. System Sciences*, pp. 220a, 2006.

[13] S. Panjwani, S. Tan, K. Jarrin, and M. Cukier, "An experimental evaluation to determine if port scans are precursors to an attack," *35th IEEE/IFIP Int'l Conf. Dependable Systems & Networks*, pp. 602-611, 2005.

[14] F. Pouget, M. Dacier, and V. Hau Pham, "Leurre.com: On the advantages of deploying a large scale distributed honeypot platform," *E-Crime and Computer Conf.*, 2005.

[15] N. Provos, J. McClain, and K. Wang, "Search Worms," *4th ACM Workshop on Recurring Malcode*, 2006.

[16] R. Shreves, *The 2008 Open Source CMS Market Share Report*, Water & Stone http://www.waterandstone.com, 2008.