Exploring Hacker Assets in Underground Forums

Sagar Samtani, Ryan Chinn, Hsinchun Chen
Department of Management Information Systems
Tucson, AZ 85721, USA
sagars@email.arizona.edu, rmc1@email.arizona.edu, hchen@eller.arizona.edu

***A*bstract- Many large companies today face the risk of data breaches via malicious software, compromising their business. These types of attacks are usually executed using hacker assets. Researching hacker assets within underground communities can help identify the tools which may be used in a cyberattack, provide knowledge on how to implement and use such assets and assist in organizing tools in a manner conducive to ethical reuse and education. This study aims to understand the functions and characteristics of assets in hacker forums by applying classification and topic modeling techniques. This research contributes to hacker literature by gaining a deeper understanding of hacker assets in well-known forums and organizing them in a fashion conducive to educational reuse. Additionally, companies can apply our framework to forums of their choosing to extract their assets and appropriate functions.**

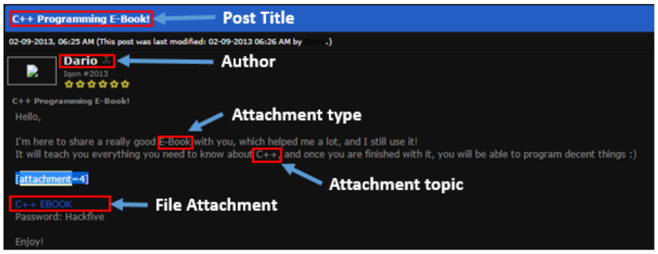
Keywords- *cybersecurity; hacker assets; topic modeling*

I.INTRODUCTION

As computers and technology become more widespread in society, cybersecurity is becoming an important concern for individuals and organizations alike. Many large companies today face the risk of data breaches via malicious software, thus compromising their business. Recent examples of such breaches include Home Depot, Target, Sony, and Xbox Live. These types of attacks are usually executed using hacker assets. Hackers in underground forums often trade and sell such assets to gain reputation [1] [6]. For example, the source code used in the Target attack (BlackPOS) was available for sale in underground markets before the attack was conducted [12].

Hacker assets come in different forms. Three of the most commonly used assets are attachments, source code, and tutorials. Figures 1, 2, and 3 illustrate each type of asset. Attachments are files attached to forum postings. These files could be books, videos, pictures, executables, tools, or various other programs. Source code is code written in a programming language embedded in forum postings. Unlike an attachment, it has not yet been compiled but is instead more raw and incomplete (cannot be executed independently without other chunks of code). Code in hacker forums can be SQL injection code, Java development code or simply examples of general programming. Tutorials, usually appearing as postings within a forum, are "how to's" or "guides" about a particular subject, designed to help others the topic. For example, a tutorial may instruct other members on how to conduct a phishing attack.

Overall, the applications of these assets range from educating other hackers on general topics, to tools and tutorials specifically designed to cause harm to other systems.



C++ Programming E-Book!

02-09-2013, 06:25 AM (This post was last modified: 02-09-2013 06:26 AM by Dario.)

Dario
Iqon #2013

C++ Programming E-Book!

Hello,

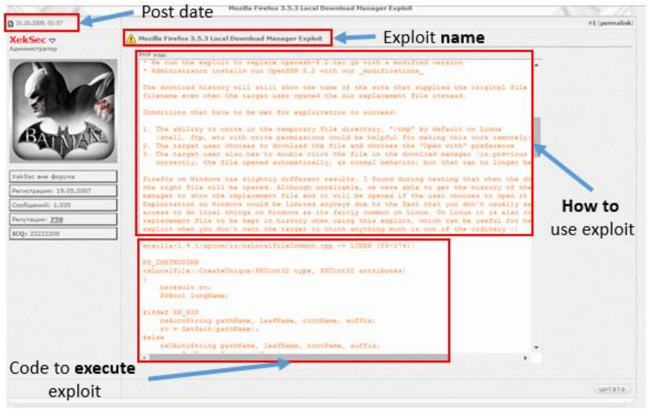
I'm here to share a really good E-Book with you, which helped me a lot, and I still use it!
It will teach you everything you need to know about C++, and once you are finished with it, you will be able to program decent things :)

[attachment=4]

C++ EBOOK
Password: Hackfive

Enjoy!

Figure 1. Forum member providing an attachment of a C++ e-book for the community. Such postings typically describe the functions/purpose of the attachment.



Mozilla Firefox 3.5.3 Local Download Manager Exploit

31.10.2009, 01:57
XekSec
Администратор
XekSec вне форума
Регистрация: 19.05.2007
Сообщений: 1,035
Репутация: 759
ICQ: 22222208

#1 (permalink)

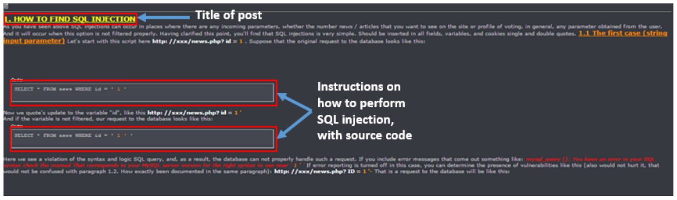
Mozilla Firefox 3.5.3 Local Download Manager Exploit

PHP код:

```
* We run the exploit to replace openssh-5.2.tar.gz with a modified version
* Administrator installs our OpenSSH 5.2 with our _modifications_

The download history will still show the name of the site that supplied the original file
filename even when the target user opened the our replacement file instead.

Conditions that have to be met for exploitation to succeed:

1. The ability to write in the temporary file directory, "/tmp" by default on Linux
   (shell, ftp, etc with write permissions could be helpful for making this work remotely)
2. The target user chooses to download the file and chooses the "Open with" preference
3. The target user also has to double click the file in the download manager (in previous
   correctly, the file opened automatically, as normal behavior; but that can no longer be

Firefox on Windows has slightly different results. I found during testing that when the do
the right file will be opened. Although unreliable, we were able to get the history of the
manager to show the replacement file and it will be opened if the user chooses to open it
Exploitation on Windows would be limited anyways due to the fact that you don't usually se
access to do local things on Windows as its fairly common on Linux. On Linux it is also co
replacement file to be kept in history when using this exploit, which can be useful for he
exploit when you don't want the target to think anything much is out of the ordinary :)
```

```
mozilla-1.9.1/xpcom/io/nsLocalFileCommon.cpp -> LINES (55-174):

NS_IMETHODIMP
nsLocalFile::CreateUnique(PRUint32 type, PRUint32 attributes)
{
    nsresult rv;
    PRBool longName;

#ifdef XP_WIN
    nsAutoString pathName, leafName, rootName, suffix;
    rv = GetPath(pathName);
#else
    nsCAutoString pathName, leafName, rootName, suffix;
```

Figure 2. Forum posting with embedded source code.


Title of post
Instructions on how to perform SQL injection, with source code

1. HOW TO FIND SQL INJECTION

As you have seen above SQL injections can occur in places where there are any incoming parameters, whether the number news / articles that you want to see on the site or profile of voting, in general, any parameter obtained from the user. And it will occur when this option is not filtered properly. Having clarified this point, you'll find that SQL injections is very simple. Should be inserted in all fields, variables, and cookies single and double quotes. 1.1 The first case (string input parameter) Let's start with this script here http: //xxx/news.php? id = 1 . Suppose that the original request to the database looks like this:

Code:

```
SELECT * FROM news WHERE id = ' 1 '
```

Now we quote's update to the variable "id", like this http: //xxx/news.php? id = 1 '
And if the variable is not filtered, our request to the database looks like this:

Code:

```
SELECT * FROM news WHERE id = ' 1 ' '
```

Here we see a violation of the syntax and logic SQL query, and, as a result, the database can not properly handle such a request. If you include error messages that come out something like: mysql_query (): You have an error in your SQL syntax check the manual That corresponds to your MySQL server version for the right syntax to use near ' 1 ' ' If error reporting is turned off in this case, you can determine the presence of vulnerabilities like this (also would not hurt it, that would not be confused with paragraph 1.2. How exactly been documented in the same paragraph): http: //xxx/news.php? ID = 1 '- That is a request to the database will be like this:

Figure 3. Forum posting of a member providing a tutorial on performing a SQL Injection. Keywords such as "How to" were used to find this tutorial

Should hackers gain insight into systems and applications at an organization, they can identify vulnerabilities and potentially exploit them with assets found in forums. Researching hacker assets within underground communities can:

- Help identify the tools which may be used in a cyberattack
- Provide knowledge on how to implement and use such assets
- Assist in organizing tools in a manner conducive to ethical reuse and education

Therefore, we are motivated to research and develop a general framework aimed at determining the application and purpose of tutorials, source code, and attachments in hacker communities as well as organizing these assets in a manner to facilitate educational reuse. The main contributions of this study includes an increased understanding of hacker forum assets; a general semi-automatic framework to identify and topically classify hacker forum assets; organization of hacker assets; and the identification of potential threats in popular hacker forums.

II. LITERATURE REVIEW

To form the basis of this research, we first reviewed hacker community research. This research provides contextual insight on hacker behaviors and hierarchies in forums as well as an understanding of the key hackers and the types of assets they create and distribute. We reviewed three sub-areas- Hacker community behaviors/interactions, focused on hacker network composition and interactions; Key hackers within communities, focused on identifying the most prolific and/or influential hacker leaders; Hacker forum contents, focused on analyzing the content, services, and information in hacker forums.

A. Hacker community behaviors and interactions

The focus of hacker community research is on understanding the social network and interactions of members in hacker communities. This work primarily uses manual explorations for qualitative analysis in Russian, English and German hacker forums [5] [11] [12] [20] [24]. Such methods have shown that the majority of participants in hacker forums are unskilled. A moderate sized group is semi-skilled, and a small percentage is highly skilled [11]. Assets flow from the skilled, members down to the less skilled members. Such tiers of hackers exist in English, Russian and German hacker forums [11] [12] [19]. Forums often rank their members, usually on the frequency of member contributions (often in the form of assets) to the community [20]. In such forum structures, the technical competency and cost for those wishing to conduct attacks or gain information about a particular topic is relatively low [5].

Key hacker literature uses various approaches to identify key hackers and their characteristics in English, Russian and Chinese forums. For example, research has identified the top and lowest malware carding sellers in a Russian forum by using snowball sampling to find malware and carding threads, classifying them using maximum entropy, and applying deep learning-based sentiment analysis [14]. In another study, Interaction Coherence Analysis (ICA) and clustering methods have been used to identify that about 12% of the hacker forum ic0de are “technical enthusiasts” who embed source code and attachments into their postings the most [1].

The embedding of source code, attachments and other content that advances knowledge in hacker communities also plays significant role in determining a hackers’ reputation [6]. While the reputable, highly skilled subset of members often create assets, intermediaries (moderately skilled members) are usually the primary distributors of these assets [2].

Hacker forum contents studies have generally focused on understanding the contents of hacker communities by using interviews with subject matter experts and manual exploration of underground hacker forums and black markets. Such methods have revealed that a variety of items can be found in underground communities. Payloads, full services and credit card information is often available on hacker black markets [2]. In addition, hosting services, currency sources and mobile devices in underground communities often facilitate cybercriminal activities [10]. Furthermore, a variety of older malware code is available in hacker forums free of charge [5].

As previously mentioned, understanding the topics and purpose of source code is a non-trivial task compared to attachment and tutorial postings. Thus, source code classification and source code topic extraction techniques are reviewed. Such literature helps provide insight on classifying and extracting topics from source code, and can be adopted to source code found in hacker forums for better code organization.

B. Source code analysis methods

Source code analysis literature is reviewed to gain an understanding of how to automatically extract topics from code and how to classify code into their appropriate programming languages. Unlike attachments and tutorials in which the topics and purpose can be easily extracted using topic modeling techniques, understanding source code is non-trivial. Additionally, source code provides content and structure conducive to discovering its technical implementation. This can lead to better organization of code assets, one of the aims of this study. We review two sub-areas of source code literature – source code classification and source code topic extraction.

Source code classification generally focuses on classifying the functions of code in online software repositories like SourceForge or Ibiblio into pre-defined categories such as databases, games, email or other domain specific areas [7][18][22]. This type of research is motivated by the desire for better software reuse, organization and maintenance [18].

The general strategy to classify source code is to identify a set of target classes for classification (databases, games, communications etc.), and develop a training set with sample source code from each of those classes [7] [15] [18]. This strategy is generally useful when the programming language of the code being classified is the same. The training set typically contains features which are unique to the classes which they are representing [15] [18]. A variety of classification methods have been used in this task, with Support Vector Machine (SVM) consistently having the highest performance [15] [18] [22].

While the focus in this stream is on classifying a single type of source code into known domain classes, source code can also be classified into their appropriate programming language. By collecting sample source code files in various programming languages to develop a feature set, a classifier can be trained to classify source code files into their appropriate languages. SVM classifiers using the LIBSVM package appear to be the most effective in this type of source code classification [22].

However, classifying the programming language does not identify the function or purpose the code serves, unless it is already in a pre-defined category. Determining the function of code without proper context or execution is a non-trivial task. If the source code is complete, it can be executed to identify its function or purpose. However, if source code is incomplete or not in a pre-defined category (as most code in online forums is), topic modeling techniques are often used.

This area of source code analysis focuses on extracting topics and functions of source code. Typically, such research is often applied to large software systems [17] [21]. However, the same techniques have been applied to well-known online code repositories [3] [4] [16].

The primary method to extract topics from source code is Latent Dirichlet Allocation (LDA), also known as topic modeling. LDA is a statistical technique modeling latent topics in text documents in a hierarchical fashion. While LDA is typically applied to normal text documents, literature in this stream has adapted the LDA model for source code. LDA is used in source code analysis when the target categories (i.e., games, databases, etc.) are unknown, as it often can be in online repositories [3][4][21]. Once processed, the files are run through Mallet (typically for 40 topics) and are manually labeled [4][21].

III. RESEARCH GAPS AND QUESTIONS

Based on prior literature in hacker communities and source code analysis, several research gaps have been found in both areas of literature. In hacker communities, much work has focused on hacker interactions and key hacker identification, but little work has focused on the assets found in hacker forums. Additionally, little work has focused on automatically identifying functions and topics of these assets, specifically source code, tutorials and attachments.

In source code analysis research, little work has been done on classifying or extracting topics of source code in hacker communities. The majority of these studies have been applied to online software repositories or software systems, but not source code found in hacker contexts. Based on these gaps, the following research questions are proposed for this study:

- What are the characteristics and functions of hacker assets in underground communities?
- What is the most effective language classification method for hacker source code?

This study addresses the research gaps in several ways. First, it provides for a better understanding of the functions, purposes, and key features of hacker assets. Secondly, this study identifies the technical features which make up the source code tools in forums. Finally, it extends existing source code literature to online hacker forums.

IV. RESEARCH TESTBED

Five hacker forums are identified for collection and analysis; they are listed in Table I. These forums were selected for several reasons. First, members in these forums have the ability to embed source code and attach files to their postings. Second, these forums are known to contain a variety of tools for members. Third, these forums can be accessed without any payment or invitation, thus making the base of potential users large. Finally, these forums are well known and have minimal downtime, allowing for optimal collection and analysis.

TABLE I. Research Testbed

Language	Forum	Date Range	# of attachments	# of source code snippets *	# of postings
English	hackfive	1/24/2013-12/29/2014	3	1,679	16,674
English	hackhound	12/2/2012-12/30/2014	151	360	7,545
English	icode	1/1/2009-2/12/2014	519	4,640	13,491
Russian	Exploit.in	1/1/2007- 1/1/2015	734	2,112	47,053
Russian	zloy	1/1/2005- 1/2/2015	1,844	6,153	486,870
TOTAL:	5	1/1/2005-1/2/2015	3,251	14,944	671,633

These forums were collected via automated methods. A web crawler routed through the Tor network was used to download the web pages. Regular expressions were then used to parse the web pages and store attributes of interest (post, author and thread information) into a MySQL database.

V. RESEARCH DESIGN

Our hacker asset analysis framework comprises of four main components: snowball sampling, data preprocessing, asset analysis and evaluation (Figure 4).

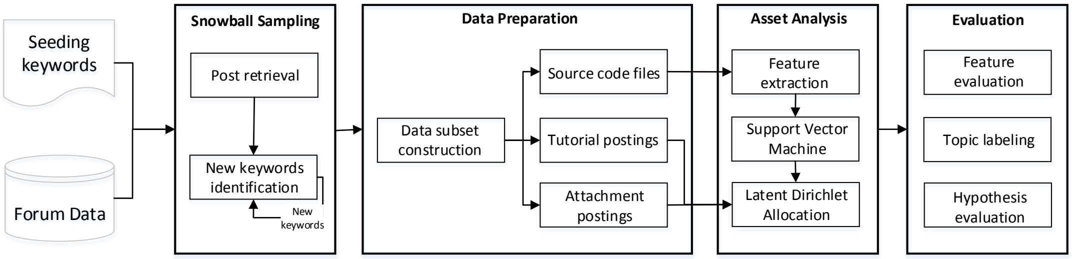


Figure 4. Research Design

A. Snowball sampling and post retrieval

The first step in this framework is to retrieve the source code, attachment and tutorial postings. SQL queries are used to retrieve code and attachment postings. However, snowball sampling is employed to retrieve tutorial postings from the database. This is similar to prior research in which snowball sampling is used to retrieve malware and carding threads [14]. Starting with a set of seeding keywords such as “how to” or “guide” or “tutorial,” postings are iteratively retrieved. The user names in those postings are used as new keywords to extract other postings.

B. Data Preparation

Once all of the postings have been retrieved, they are then split into three subsets: source code postings, tutorial postings, and postings with attachments. All duplicate postings with each subset are removed. Thread titles are considered to be part of each of the postings as they can help provide more context for topic modeling. If a posting contains any embedded source code, it is placed in the source code subset. All other postings are placed in their respective categories.

C. Asset analysis

Once in their appropriate subsets, LDA is used to understand the topic characteristics of hacker assets. LDA is performed in an identical manner on both the attachment and tutorial postings. Attachment and tutorial postings in hacker forums are typically descriptive of the type of file and the instructions attached in the posting. Thus, the topics and applications of these assets are relatively clear. Each subset is treated as its own corpus, with each posting being a document. Porter’s stemming algorithm is used to unify words to their common root (i.e., attacking and attack get stemmed to attack) and a stop-words list is used to filter generic terms in the posting. Once stemmed and filtered, 40 topics are extracted from each subset using a Mallet based tool, consistent with prior literature. These topics are then manually labeled and tabulated.

While attachment and tutorial postings can be evaluated in similar fashions, analyzing the topics of source code is a non-trivial task. As with the attachments and tutorial subsets, the source code postings subset is considered to be a corpus and all the postings are considered to be documents. Consistent with prior literature, we leave the comments in to help provide context for topic modeling. In addition, we treat post content as comments, as it often contains information about the purpose of the embedded code. However, unlike attachment and tutorial postings, source code postings contain content that is not part of natural language. As a result, the manner in which they are processed for LDA is slightly different than the tutorials and attachments.

The first step in processing the source code postings for LDA is splitting the identifiers found in source code to meaningful sub-words which can be better interpreted. For example, the identifier DATA_AUTH_RESPONSE is split into DATA, AUTH and RESPONSE. Once split, Porter’s stemming algorithm is then applied to unify words to their root. A stop-words list is then used to filter generic terms in the posting [3][4][16][17][21]. LDA is then run for 40 topics using the same tool which was used for attachment and tutorial postings. These topics are then manually labeled and tabulated.

In addition to using LDA to find the topics of the source code postings, additional analysis is conducted to see how these source code assets are being implemented (i.e., what language is used to create the source code). Such information is useful for several reasons. First, classifying the programming language allows for better code reuse, which can be useful for educational purposes. Secondly, it provides insight into the implementations of particular types of assets, specifically into the key features necessary to create such assets. Finally, the classification of source code into their appropriate programming languages facilitates better organization.

To perform source code classification, a classifier must first be trained using sample source code files. From observing the types of source code postings in forums, 10 classes were selected for classification: Java, Python, C/C++, HTML, PHP, Delphi, ASP, SQL, Ruby and Perl. One hundred code files for each language were used from online repositories such as SourceForge and GitHub to train the classifier. These files were manually selected based on their uniqueness to each of the languages. Files were typically selected from libraries which are exclusive to a language (i.e., JSoup appears only in Java), as that increases the effectiveness of the classifier.

After the files are selected and the classifier trained, the source code files within each of the topics created by LDA are then used as test-beds for the trained classifier. The classifier was implemented using the LIBSVM package as provided by RapidMiner. After each topic has each of its files classified into their appropriate languages, the outputted features are then tabulated and evaluated.

D. Evaluation

The evaluation section of this framework consists of three components- manual topic labeling, feature evaluation and hypothesis evaluation. The outputted topics of the LDA model are manually evaluated and labeled. In the case of the source code, once the topics are labeled and their respective files are classified, the key features of those topics/languages are tabulated.

To evaluate the hypothesis, an experiment was conducted comparing the performance of four classifiers: Support Vector Machine, Naïve Bayes, K-Nearest Neighbor and decision tree. These classifiers are commonly used in past literature.

VI. RESEARCH HYPOTHESES

Past literature has indicated that SVM outperforms other classifiers in classifying source code into their appropriate languages. Therefore, to test the effectiveness of our framework, we propose the following research hypothesis:

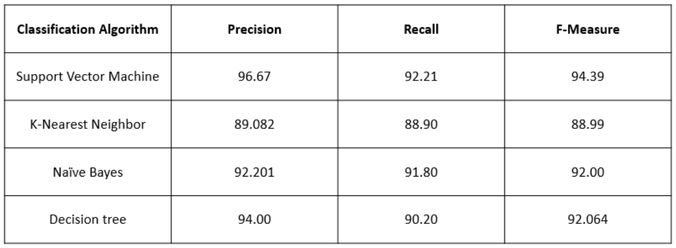
H1: Support Vector Machine will outperform K-Nearest-Neighbor, Decision Tree and Naïve Bayes in categorizing source code in terms of precision, recall and F-Measure.

VII. FINDINGS AND RESULTS

A. Evaluation

An experiment was conducted to test the effectiveness of the classification component of our framework. In the experiment, we compared our system (SVM) against other widely used classifiers: k-Nearest Neighbor, Naïve Bayes, and Decision Tree. All the classifiers were trained using the same set of 100 source code files. In our experiment, SVM outperformed K-Nearest Neighbor, Naïve Bayes, and Decision Tree in terms of precision, recall, and F-Measure, supporting H1. This is also consistent with prior literature. Table II shows the results of our experiment.

TABLE II. SOURCE CODE CLASSIFICATION PERFORMANCE COMPARISON



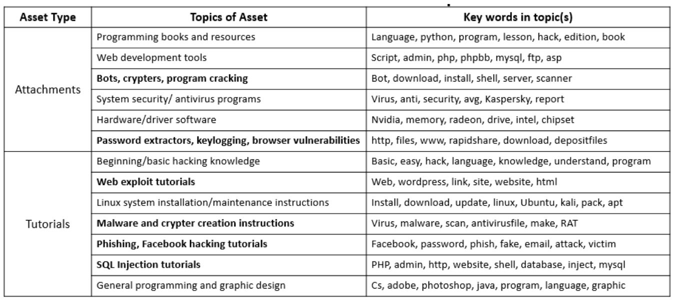
Classification Algorithm	Precision	Recall	F-Measure
Support Vector Machine	96.67	92.21	94.39
K-Nearest Neighbor	89.082	88.90	88.99
Naïve Bayes	92.201	91.80	92.00
Decision tree	94.00	90.20	92.064

B. Topics of hacker assets

By using LDA on attachments and tutorials, we are able to find several interesting results. There are several types of attachments available that are designed to cause direct harm to other machines including the Dirt Jumper DDoS attack, keyloggers and crypters. However, many of the attachments are general, non-threatening resources for other community members to use (books, system security tools, hardware/ software).

Tutorials appear to be the primary method of providing community members with resources to carry out malicious attacks. Some of the more interesting tutorials pertain to SQL Injections, web exploits, creating malware, and crypters. Tutorials are particularly interesting as they allow for immediate, step-by-step instructions on how to carry out a particular task. Table III highlights the topics and their associated keywords which define that topic. We have selected only the topics which are the most interesting or represent broad hacker tutorials/ attachments.

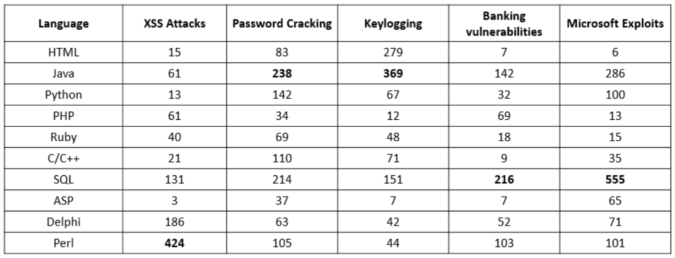
TABLE III. TOPICS OF ATTACHMENTS AND TUTORIALS



Asset Type	Topics of Asset	Key words in topic(s)
Attachments	Programming books and resources	Language, python, program, lesson, hack, edition, book
	Web development tools	Script, admin, php, phpbb, mysql, ftp, asp
	Bots, crypters, program cracking	Bot, download, install, shell, server, scanner
	System security/ antivirus programs	Virus, anti, security, avg, Kaspersky, report
	Hardware/driver software	Nvidia, memory, radeon, drive, intel, chipset
	Password extractors, keylogging, browser vulnerabilities	http, files, www, rapidshare, download, depositfiles
Tutorials	Beginning/basic hacking knowledge	Basic, easy, hack, language, knowledge, understand, program
	Web exploit tutorials	Web, wordpress, link, site, website, html
	Linux system installation/maintenance instructions	Install, download, update, linux, Ubuntu, kali, pack, apt
	Malware and crypter creation instructions	Virus, malware, scan, antivirusfile, make, RAT
	Phishing, Facebook hacking tutorials	Facebook, password, phish, fake, email, attack, victim
	SQL Injection tutorials	PHP, admin, http, website, shell, database, inject, mysql
	General programming and graphic design	Cs, adobe, photoshop, java, program, language, graphic

The majority of the source code found in these forums was not related to specific attacks, but instead mostly about general topics. However, several interesting topics relating to various types of attacks were found. The one of significant interest is a topic pertaining to banking vulnerabilities. Using the SVM classifier, we were able to identify the methods in which this topic implemented. The majority of the source code used in this topic fall under SQL, suggesting that these attacks may be targeted towards the databases of banks. Table IV provides a complete breakdown of the most interesting topics and their primary method of implementation.

TABLE IV. SOURCE CODE TOPICS AND IMPLEMENTATION METHODS



Language	XSS Attacks	Password Cracking	Keylogging	Banking vulnerabilities	Microsoft Exploits
HTML	15	83	279	7	6
Java	61	**238**	**369**	142	286
Python	13	142	67	32	100
PHP	61	34	12	69	13
Ruby	40	69	48	18	15
C/C++	21	110	71	9	35
SQL	131	214	151	**216**	**555**
ASP	3	37	7	7	65
Delphi	186	63	42	52	71
Perl	**424**	105	44	103	101

VIII. CONCLUSION AND FUTURE DIRECTIONS

This research aims to understand the functions and characteristics of assets in hacker forums by applying classification and topic modeling techniques. Specifically, we focus on extracting the topics of source code, attachments and tutorials. This research contributes to hacker literature by gaining a deeper understanding of hacker assets in well-known forums and organizing them in a fashion conducive to educational reuse. Source code literature is extended by applying state of the art classifiers to a new area, hacker code.

This research also has practical contributions. Companies can apply this framework to forums of their choosing to extract the types of assets which are present in it. Using such information, companies can implement appropriate defenses for their systems. Finally, this framework facilitates easier search and browsing of hacker assets, which can be used for educational purposes. Tutorials provide immediate education, as they are step-by-step instructions for a task (i.e., phishing).

Future work can expand this research to increase understanding of hacker assets. In addition to increasing the variety of forums, future work can look at the specific members creating and disseminating such assets by using social network analysis techniques. This work can also be expanded by introducing a temporal component to identify the prevalence of particular assets over time. This type of information would be especially useful when trying to determine emerging assets.

ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation (DUE-1303362 and SES-1314631).

REFERENCES

[1] Abbasi, A., Li, W., Benjamin, V., & Hu, S. (2013). Descriptive Analytics: Examining Expert Hackers in Web Forums. *IEEE Joint Intelligence and Security Informatics Conference,* 56-63.

[2] Ablon, L., Libicki, M. C., & Golay, A. a. (2014). Markets for Cybercrime Tools and Stolen Data: Hacker's Bazaar. National Security Research Division.

[3] Baldi, P. F., Lopes, C. V., Linstead, E. J., & Bajracharya, S. K. (2008). A theory of aspects as latent topics. ACM SIGPLAN Notices, 43, 543.

[4] Barua, A., Thomas, S. W., & Hassan, A. E. (2012). What are developers talking about? An analysis of topics and trends in Stack Overflow. Empirical Software Engineering (pp. 1–36).

[5] Chu, B., Holt, T., & Ahn, G. (2010). Examining the Creation, Distribution, and Function of Malware On Line. Retrieved from https://www.ncjrs.gov/App/Publications/abstract.aspx?ID=252143

[6] Benjamin, V. A., and Chen. H. (2012). Securing Cyberspace: Identifying Key Actors in Hacker Communities. IEEE International Conference on Intelligence and Security, 24-29.

[7] Benjamin, V. A., & Chen, H. (2013). Machine Learning for Attack Vector Identification in Malicious Source Code. IEEE Intelligence and Security Informatics, 21–23.

[8] Décary-Hétu, D., & Dupont, B. (2013). Reputation in a dark network of online criminals. Global Crime, 14, 175–196.

[9] Décary-Hétu, D., & Dupont, B. (2013). Reputation in a dark network of online criminals. Global Crime, 14, 175–196.

[10] Gad, (2014). Markets for Cybercrime Tools and Stolen Data: Hacker's Bazaar. National Security Research Division.

[11] Holt, T. J., Strumsky, D., Smirnova, O., & Kilger, M. (2012). Examining the Social Networks of Malware Writers and Hackers. International Journal of Cyber Criminology, 6(1), 891–903.

[12] Holt, T. J. (2013). Examining the Forces Shaping Cybercrime Markets Online. Social Science Computer Review, 31(2), 165–177

[13] Kitten, Tracy. "Target Malware: Exploring the Origins." Bank Info Security. N.p., 20 Jan. 2014. Web. 1 Dec. 2014. <http://www.bankinfosecurity.com/interviews/intelcrawler-i-2161>.

[14] Li, W., & Chen, H. (2014). Identifying Top Sellers In Underground Economy Using Deep Learning-based Sentiment Analysis. IEEE Joint Intelligence and Security Informatics Conference, 64-67.

[15] Linares-Vásquez, M., McMillan, C., Poshyvanyk, D., & Grechanik, M. (2014). On using machine learning to automatically classify software applications into domain categories. Empirical Software Engineering (Vol. 19, pp. 582–618).

[16] Linstead, E., Lopes, C., & Baldi, P. (2008). An application of Latent Dirichlet Allocation to analyzing software evolution. Proceedings - 7th International Conference on Machine Learning and Applications, ICMLA 2008, 813–818.

[17] Maskeri, G., Sarkar, S., & Heafield, K. (2008). Mining business topics in source code using latent dirichlet allocation. Isec, 113–120.

[18] McMillan, C., Linares-Vásquez, M., Poshyvanyk, D., & Grechanik, M. (2011). Categorizing software applications for maintenance. IEEE International Conference on Software Maintenance, ICSM, 343–352.

[19] Motoyama, M., McCoy, D., Levchenko, K., Savage, S., & Voelker, G. M. (2011). An Analysis Of Underground Forums. Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference - IMC '11, 71.

[20] Radianti, J. (2010). A study of a social behavior inside the online black markets. Proceedings - 4th International Conference on Emerging Security Information, Systems and Technologies, SECURWARE 2010, 189–194.

[21] Tian, K., Revelle, M., & Poshyvanyk, D. (2009). Using Latent Dirichlet Allocation for Automatic Categorization of Software. 6th IEEE International Working Conference on Mining Software Repositories, 163–166.

[22] Ugurel, S., Krovetz, R., 'Z, C. L. G., Pennock, D. M., Glover, E. J., & Zha, H. (2002). What's the code? automatic classification of source code archives. Proceedings of the Eighth Acm Sigkdd International Conference on Knowledge Discovery and Data Mining, 2, 632 – 638.

[23] Wang, T., Wang, H., Yin, G., Ling, C. X., Li, X., & Zou, P. (2013). Mining software profile across multiple repositories for hierarchical categorization. IEEE International Conference on Software Maintenance, ICSM, 240–249.

[24] Yip, M. (2011). An Investigation into Chinese Cybercrime and the Applicability of Social Network Analysis. ACM Web Science Conference.