Open Access License Notice

An Analysis of Security Competitions for A Beginner's Guide

Qijun Gu
qijun@txstate.edu

Tanner J. Burns
tjb102@txstate.edu

Samuel C. Rios
scr3@txstate.edu

Thomas K. Jordan
tkj15@txstate.edu

Texas State University
San Marcos, TX 78666

Trevor Underwood
tunderwood@netspend.com

Netspend Corporation
Austin, TX 78768

Abstract - Security competitions are emerging as a new approach in security education and professional training. At universities, security competitions are gradually introduced into Computer Science curriculum to attract more students into the security area and prepare them for a career in the security field. The benefits of competition-based education were recognized in many studies. However, there are still many challenges for beginners to participate in the competitions. To help beginners to study and participate, this paper analyzed thousands of competition problems in over a hundred security competitions in the past three years. This paper identifies several important characteristics of the security competitions, including the main security areas and the fundamental knowledge and skills to solve problems in these areas. This

paper presents the findings as guidance to beginners so that they can find their interested areas to study and practice.

Keywords

Security Competition, Catch The Flag, Security Education, Competition Analysis, Competition Guide

1 INTRODUCTION

Over the past several years, security competitions, such as Defcon [3], CCDC [4] and many capture-the-flag (CTF) competitions [1,7,16], are emerging as a popular method of attracting promising students into security education and careers. The competitions were sponsored by either industry or government agencies and held around the United States and worldwide for high school students, college students and even professionals. They aimed to train the next generation of security professionals using hands-on competitions and to enhance the interests of a security career among the students. They have generally been seen as great methods for security education, training and recruitment.

There were many efforts to incorporate security competitions and similar practices into security education [6,8,11,13,15]. Educators and researchers have also collected empirical data to study the effectiveness of security competitions [5,14] and found that security competitions offered valuable learning experiences for computer science students as well as students in many other disciplines, such as criminology and criminal justice. The competitions improved the students' hands-on skills as well as their understandings of cyber attacks and defenses.

Although security competitions are beneficial to students interested in security, it is recognized that there are inevitable challenges for beginner students to get involved in security competitions [10,12,17]. Often, beginners got frustrated and discouraged because they were unable to solve problems. Even though they may

have studied background security knowledge in class, they still lack sufficient skills in coding, networking and system administration, are not proficient in using security tools, and do not know specific security flaws. Beginners need to overcome many obstacles technically and psychologically to truly build confidence and gain benefits from the competitions.

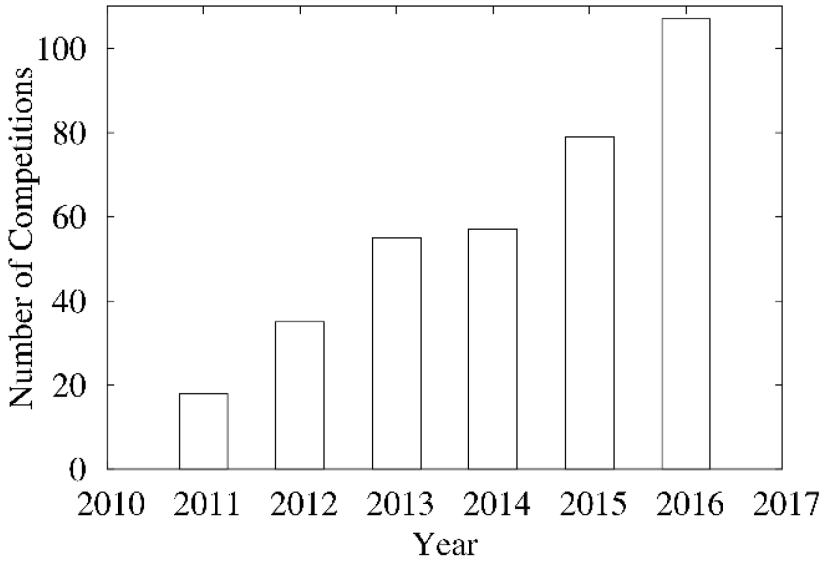
In our experience, when beginner students were introduced to a security competition for the first time, they were often lost on what to study. Security competitions typically include a variety of security problems that need a unique set of knowledge and skills to solve. When beginners are immersed with these problems, they often do not have a clue on which problem they should start with. They are often confused by many specific security areas and cannot decide which areas best match their interests and strengths.

To help beginners gain confidence and start participating in competitions, the main goal of this paper is to give a clear picture to beginners that shows (1) the main security areas in competitions, (2) the main characteristics of competitions and competition problems, and (3) the main security knowledge and skill sets necessary to solve some common problems. With this guidance, beginners can choose specific subjects to study and prepare in order to become capable of solving a few problems in their initial attempts.

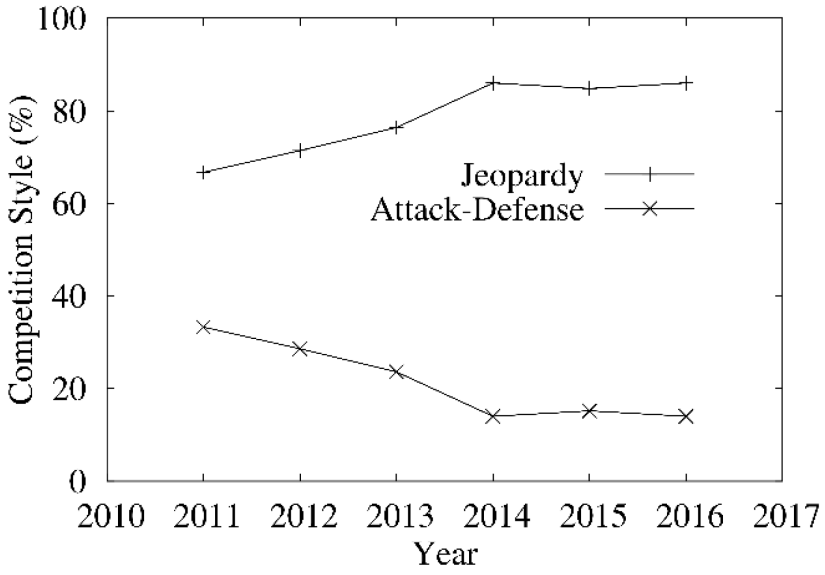
The contribution of this paper is established on a comprehensive study and analysis of over 3000 competition problems used in 160 security competitions held during 2014 to 2016. To our best knowledge, this is the first study to collect, analyze and characterize a vast amount of problems of past competitions. After analysis, we summarized the nature of the security competitions in recent years, identified six common security problem categories, and identified the mostly used knowledge and skill sets in the security areas. With these findings, we thus make our beginner's guide that recommends some must-have skill sets for beginners to study.

In the rest of the paper, we first describe our methods of collecting and analyzing data in Section 2. We present the main characteristics of the security problems in

Section 3. Then, we present guidance for beginners to study in Section 4. Finally, we discuss the related work on using security competitions in computer science education in Section 5 and conclude in Section 6.



(a) Number of Competitions Over Years



(b) Competition Styles

Figure 1: Characteristics of Past Competitions

2 DATA COLLECTION AND ANALYSIS OF COMPETITIONS

2.1 Security Competitions

Many security competitions have taken place around the world in recent years. We collected the data of the past competitions during 2011 and 2016 from the archives of CTFtime.org [1]. Figure 1(a) shows that the number of competitions had linear growth in the past six years. Security competitions have clearly been attracting more and more hosts and players in industry and academy for not only training security professionals but also business-involved activities, such as recruiting, advertising and so on.

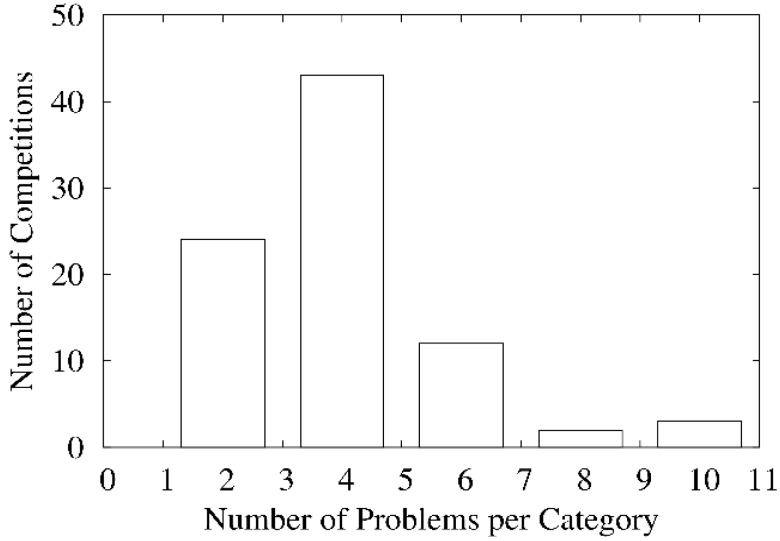
Security competitions are often categorized as jeopardy style, where players use offensive techniques to solve security problems, defense style, where players need

to defend their vulnerable services, and attack- defense style, where players need to take both offensive and defensive actions against other players. We analyzed the competition styles based on the data from CTFtime.org. We noticed that the data only has the jeopardy style and the attack-defense style competitions, and does not include any defense-only competitions. Figure 1(b) shows that more jeopardy style competitions were emerging in the past years. The numbers of the attack-defense style competitions were stable in the range of six to thirteen every year.

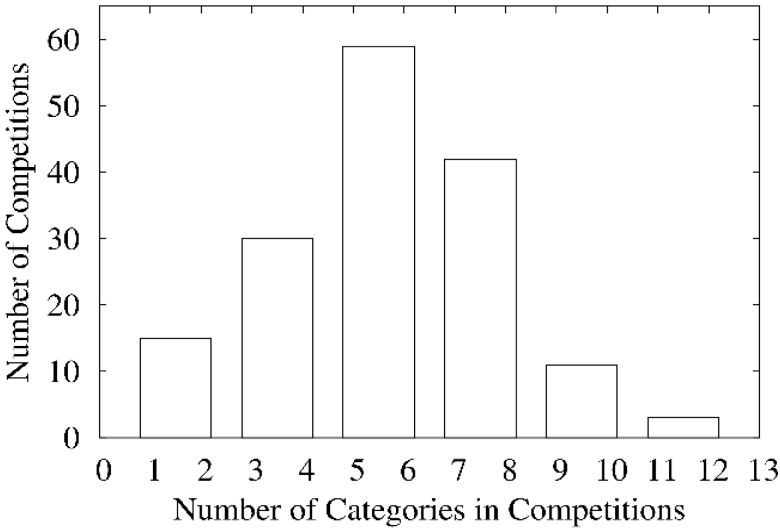
2.2 Security Problems and Categories

To better understand the types of competition problems and the associated skills that are required to solve them, we collected and analyzed not only security problems but also their solutions. Many players voluntarily posted their solutions as "writeups" that provide step-by-step solutions with commentary about their thought processes. The writeups are excellent resources for beginners to study and follow. Beginners can find the writeups on Github [2], CTFtime [1], players' personal blogs or websites, and so on. After comparing these sources, we chose to collect writeups from Github [2] due to the larger quantity and better quality of the submitted writeups. For many problems, we found multiple writeups to compare their solutions.

We collected and analyzed the writeups posted for security competitions in 2014, 2015 and 2016. There were very limited writeups before 2014 on Github, and they were scattered over other sources. In total, we collected 3598 security problems of 160 security competitions. Because the writeups are completely volunteer-based, not all security competitions have writeups. The collected writeups cover 34 of 57 (60%) security competitions in 2014, 56 of 79 (71%) security competitions in 2015, and 70 of 107 (65%) security competitions in 2016. We noticed that there are missing security problems across the 160 competitions. We could not verify how many problems are missing because we could not obtain the original problems from many past competitions.



(a) Histogram of Problem Counts



(b) Histogram of Category Counts

Figure 2: Characteristics of Problems and Categories

Figure 2(a) shows the histogram of the number of security problems with corresponding writeups on Github. Most competitions have about 20 or 30 security problems. Ten competitions have fewer than five problems with writeups. Upon further inspection, we found that two of the ten competitions were teaser competitions that did not have many problems. Two were for high school students that maybe did not have enough players to provide writeups. Two were some sort of easy qualification competitions. The remaining four might not have had enough players.

Most security competitions divided their security problems into a variety of categories. Similar problems were grouped in the same category. However, the competitions did not always name the categories in the same way. We analyzed the 909 category names used in the 160 competitions. To avoid duplication, we combined similar category names. For example, the categories "pwn", "pwnable" and "pwning" were combined to "pwn". Then, we reduced them to 77 unique category names in these competitions. We found six top category names: "crypto", "web", "reverse", "forensic", "pwn" and "misc". They represent the major categories of security problems and security areas in competitions. Often, the problems of the other categories overlap with the six categories. For example, the problems of "exploit" often belong to either "pwn" or "web", and the problems of "binary" often belong to either "reverse" or "pwn".

To make our analysis more concise and useful to beginners, we kept the six categories. We read and analyzed the writeups of the problems of the other categories and reclassified them to the six categories based on the key goals of the problems. For example, some "stegano" and "recon" problems are to extract hidden or obfuscated information, and thus are re-classified to the "forensic" category. "ppc" and "trivia" problems are reclassified to the "misc" category because they do not actually address security issues. Some original "misc" problems were reclassified to the other categories because they addressed some sort of security issues.

Categories	Problems	Reclassification
Crypto	Cryptographic problems	crypto, web, network
Web	Web exploitation problems	web, exploit, recon, network, misc
Reverse	Reverse engineering problems	reverse, binary, misc
Forensic	Data extraction problems	forensic, stegano, recon, network, misc
Pwn	Exploit remote services	pwn, exploit, network, binary, misc
Misc	Coding and non-security problems	misc, trivia, ppc

Table 1: Categories and Reclassification

Table 1 shows the six categories used in this paper and the reclassification of the categories used in the collected competitions. With fewer categories, our analysis provides more focused sets of security knowledge, skills and techniques for beginners to study.

3 ANALYSIS OF COMPETITION PROBLEMS

In this study, we read the security problems and the corresponding writeups. We discussed together the problems and the solutions from the writeups every week. The weekly discussion made us stay on a common ground for analyzing the problems and solutions. While studying the problems, we recorded the characteristics of the problems and tested the methods and tools of the solutions.

After reading all writeups of the collected 3598 problems, we further excluded 1400 problems from our analysis because their writeups are actually missing or do not have complete solutions. We then analyzed the remaining 2198 problems that have good writeups to identify and summarize the information that can help beginners to study and prepare for security competitions.

3.1 Difficulty Levels

A security competition usually assigns different points to the security problems to indicate their difficulty levels. The greater the point, the more difficult a problem is. Because each competition targets different kinds of players, there is no common criteria among the competitions to evaluate difficulty levels of the security problems. For example, some easy problems in Defcon's qualification competitions were equivalent to hard problems in competitions for high school students. Furthermore, there is no common approach to assign points to the security problems. Some competitions assigned two-digit points while others assigned three-digit or four-digit points.

Even though security problems vary, after reading many writeups, we observed that players need to possess specific technical skills, identify key methods of the problems, and use a few tools in order to solve the problems. Therefore, we define three difficulty levels below with a mixed qualitative and quantitative approach that bases the levels on the key components of the solutions from the perspectives of beginners. Referring to the points that were assigned to the problems, we re-evaluated the problems and assigned the three difficulty levels to them. We can then more fairly identify the basic and advanced skills for beginners to learn.

1. *Easy*: A problem can be solved with one or two methods and tools. A beginner can often solve the problem by themselves right after reading the writeups.

2. *Medium*: A problem can be solved with three or more methods and tools. After reading the writeups, a beginner can solve the problem with extra efforts, such as reading additional documents.

3. *Hard*: A problem can be solved with in-depth methods and sophisticated tools. A beginner can hardly understand the writeups and cannot solve the problem even after reading the writeups.

In total, more than a third of problems are easy ones. With proper studying and preparations, beginners can solve many of these easy problems in competitions and gain confidence and successful experiences. With such seed encouragement, they may gain more motivations to further their study in security.

3.2 Problem Characteristics

Based on the writeups, we identified three main characteristics of competition problems: (1) coding languages for pwn, web, reverse and misc, (2) cryptographic algorithms for crypto, and (3) data types for forensic. These characteristics reflect the minimum essential knowledge and skills that are required to solve the easy problems. These characteristics are category-specific. For example, coding language is a key characteristic of the problems in the "pwn", "web", "reverse" and "misc" categories, because players must understand the coding languages of the problems in these four categories in order to solve the problems. Coding languages are not essential for solving "crypto" or "forensic" problems, because "crypto" problems require players to understand cryptography and "forensic" problems require players to be familiar with various data types and formats.

3.2.1 Coding Languages

In many security problems, a set of programs were provided to players. The players needed to read and understand the programs and then find the key information (such as flaws) in the programs to solve the problems. Because these programs were made in a variety of coding languages, typical challenges for beginners are (i) what coding languages they need to learn and (ii) how to quickly understand a program coded in a language they do not know. We analyzed the data pertaining to coding languages to find some suggestions for the first challenge below.

For the second challenge, we think that if beginners can master a few mostly used coding languages in competitions, they can "guess" the programs in other coding languages to some extent.

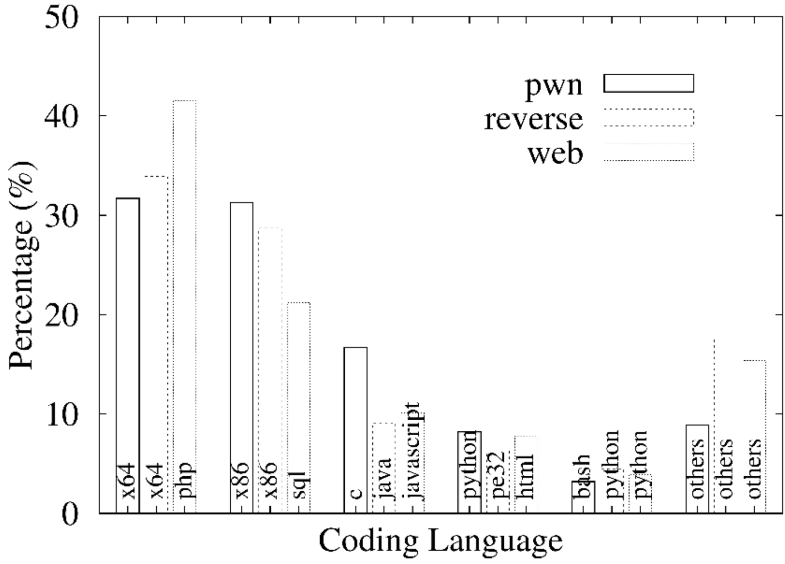


Figure 3: Coding Languages

Figure 3 shows the percentage breakdown of the coding languages in the "pwn", "reverse" and "web" categories. Only the top 5 languages of each category are listed.

In the "pwn" category, the top two coding languages are x64 assembly and x86 assembly that are used in executable binaries. Players are often required to disassemble and decompile provided binaries to find flaws and then solve the problems. When binaries were not provided in competitions, a portion of source files that built the remote services were provided to players. Then, players discovered the flaws in the provided source code to solve the problems. C and Python are the top two coding language used to build the remote services in competitions. Bash is among the top five because it is a system administration

language widely used in Unix and Linux computers. Compared with "reverse" and "web", the top five languages are dominant in the "pwn" category. Only 16 other programming languages were occasionally used in the "pwn" problems.

The "reverse" category has the same top two languages as "pwn", since most "reverse" problems were to reverse engineer executable ELF-based Linux binaries. Meanwhile, there are many other types of executable programs too. Java ranks third, because many problems asked players to analyze Java programs and Android applications that can both be decompiled to Java language. PE32 is a type of executable binaries running in Windows computers and ranks fourth. Because many competitions were hosted on Linux computers, PE32 binaries were not as popular as x64 and x86 binaries. Python programs are becoming popular too, and python bytecode was often used in the "reverse" problems. In total, we identified 26 other programming languages with lower ranks in the "reverse" category. They were mostly for specific programs, such as mobile devices (such as ARM) or game machines (such as Nintendo).

The top 2 languages in the "web" category are used on the server side. PHP-based web applications were mostly exploited through some well-known PHP's flaws in competitions. In addition, LAMP (Linux, Apache, Mysql and PHP) is a very common web framework. Many websites in competitions were setup based on this framework. Hence, PHP is the number one language in the "web" category. Because web sites are often backed with SQL databases, SQL ranks 2nd. In particular, most SQL databases were Mysql database in competitions. Javascript and HTML are the languages used on the client side and rank 3rd and 4th. But, because Node.js is emerging as a new server-side web development framework, some Javascript problems were on the security issues in Node.js. Python and Perl (not shown in the figure) rank 5th and 6th respectively, as they are also popular web development language. 21 other programming languages on either the server-side or client-side were used in the web exploitation problems.

3.2.2 Cryptographic Algorithms

A variety of cryptographic algorithms were used in the "crypto" problems. Not counting custom algorithms, we identified 45 publicly known cryptographic algorithms. Figure 4 shows the percentage breakdown of the problems based on these cryptographic algorithms and listed those that appeared in more than 1% of problems. More than a third of "crypto" problems used custom cryptographic algorithms. For these problems, the competition hosts generally provided the source code files that contain the algorithms. Players needed to find flaws in the algorithms to solve the problems. The remaining publicly known cryptographic algorithms include almost all major algorithms nowadays. Some of the algorithms have well known flaws and the others were used in a flawed way in competitions. Symmetric cryptographic algorithms appeared in about a third of problems. We identified 36 publicly known symmetric algorithms, much more than the counts of the other kinds of publicly known cryptographic algorithms. Asymmetric cryptographic algorithms appeared in about a fifth of problems, but has only ten algorithms. Among all publicly known cryptographic algorithms, RSA was the mostly used in competitions. Hash algorithms appeared in about 5.3% of problems. MD5 has well known collision issues and thus appeared in more problems than SHA1 and SHA2. The "misc" group includes the problems that were designed based on cryptographic tools, libraries and protocols.

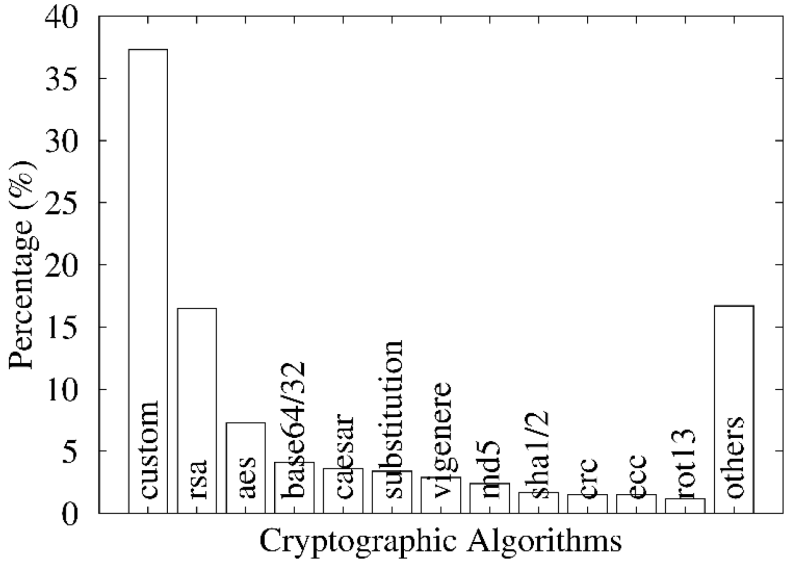


Figure 4: Cryptographic Algorithms

3.2.3 Data Types

The main objectives of problems in the "forensic" category are to extract information from various types of data files. Often, steganographic methods as well as cryptographic approaches were used together to hide information in provided data files. Among the collected problems, we identified 69 unique file formats used in the problems and divided them to 16 data types. Figure 5 shows the percentages of the problems of the top 10 data types.

Images are the number one data type, and PNG and JPG are the top two image formats. Many forensic problems embedded data in the meta information of images, concatenating multiple image files, tweaking image pixels and so on. The second most common challenges were analyzing PCAP and TCPCUMP network trace files. Players needed to follow the network traffic in the traces to find the information. The third is multimedia data, including audio, video and streaming. Information was usually hidden as a secondary track or encoded in the time or

frequency domains of the multimedia data. The other data types appeared in about one third of problems and all have their unique methods to hide information.

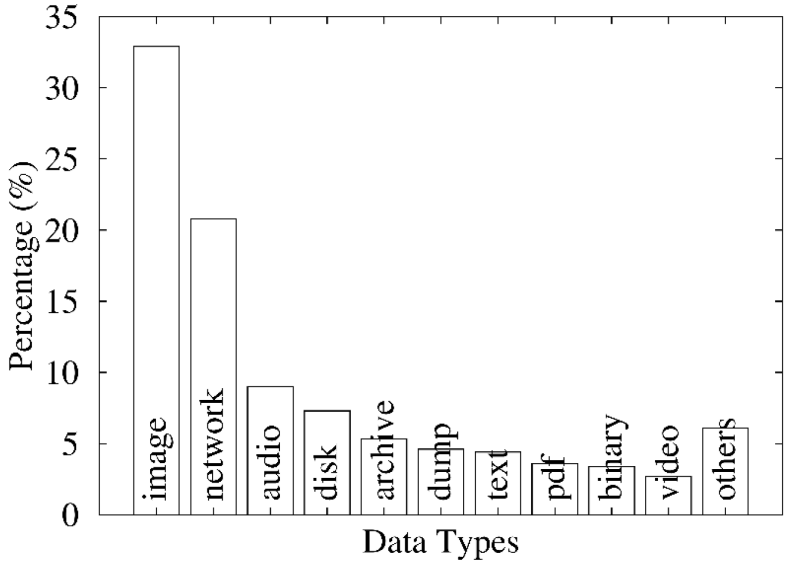


Figure 5: Data Types

4 GUIDE FOR BEGINNERS

Our analysis shows the main characteristics of competition problems. We think these characteristics not only reflect the main security issues concerned by industry and academy, but also deliver clearly to beginners what they need to study and practice for participating in security competitions. In the following, we recommend the most common knowledge and skill sets we found in our analysis as a guide for beginners. The goal of this guide is to identify many specific essential skills that are very often needed in competitions. Thereby, after learning the outlined skill sets, beginners can solve at least some easy problems in competitions. In short, beginners need to be very proficient in programming and analytics. Without the proficiency in these two skill sets, beginners cannot solve even easy problems in a timely manner.

4.1 Programming

Python appears to be the dominant programming language in the solutions of many competition problems. It is easy for beginners to learn and begin using. Beginners can quickly make some python scripts and run them to test solutions. There are a lot of supporting libraries to handle networking, web, strings, numbers, arithmetic, and various files in Python.

There are two main programming skill sets that beginners must be proficient with. The first is number, string and file manipulations, such as hexadecimal and binary conversions, string and number conversions, large number arithmetic, base64 encoding and decoding, string splitting and concatenation, and so on. Many problems provide files for analysis. Beginners should be able to open, read and analyze files in a programmable way. The second is network programming that is required to interact with remote servers in many competitions problems. Beginners shall be able to do socket programming, preferably with some well-known libraries, such as pwnlib. Thereby, beginners are able to create services, make and send arbitrary packets to remote servers, and process packets sent from remote servers. To assist with network programming, beginners should be proficient with common networking tools, such as netstat, netcat and so on.

4.2 Analytics

Analytics are mostly category-specific. In the following, we summarize the major analytic techniques according to the five security categories in Table 1 (excluding the "misc" category). Beginners are not expected to learn absolutely everything that is incorporated into all of the competitions. Rather, they should focus on what they are interested in and develop the skill sets accordingly.

Reverse: As discussed in Section 3.2.1, players often need to disassemble and decompile the executable programs in x64, x86 or Java bytecode to find flaws and then solve the problems. The executable programs are usually decompiled to source code in C or Java. Hence, beginners need to understand four coding languages: x64 assemly, x86 assembly, C and Java. In addition, beginners need to proficiently use

hex tools, disassemblers and decompilers, such as Hexdump, GDB, Hopper and IDA Pro, so that they can figure out the programming logic in the binaries.

Pwn: The problems of this categories usually requires reverse engineering to enumerate the target. Hence, in addition to the analytic techniques of the "reverse" category, beginners need to study how to exploit flaws in remote servers. Typical exploitable flaws include buffer overflow, heap overflow, format string, returnoriented programming, etc. Beginners also need to study how to make exploiting packets that exploit the flaws to attack and control the remote servers with network programming. Because the problems of the "pwn" category require more techniques, they are typically harder than the problems in the other categories. Beginners should consider "pwn" as an advanced level to "reverse".

Web: The web problems require knowledge of web technologies on both server side and client side. On the server side, beginners should familiarize themselves with PHP language, SQL language, and MySQL database, and then understand how to launch SQL injection attacks. On the client side, beginners need to study Javascript and HTML to understand how web pages are dynamically generated and rendered on the client side. Furthermore, beginners should be proficient with the use of CURL and web development tools built in most web browsers to inspect web pages and web traffics. Beginners need to inspect and manipulate cookies, sessions, URLs, form data, JSON data and web agents on the client side.

Crypto: Based on Figure 4, beginners need to master a few cryptographic algorithms. For asymmetric cryptography, beginners need to make RSA encryption and decryption programs or use exiting RSA tools and libraries. Beginners also need to study and be able to exploit a few common RSA implementation flaws, such as weak public keys and Coppersmith's attack. For symmetric cryptography, beginners need to study AES-ECB encryption and crack it when it is used in an insecure manner. Usually, easy crypto problems were based on Caesar cipher, substitution cipher, Vigenere cipher, and XOR operators. Hence, beginners should study them. In hash, beginners need to study how to perform hash reverse lookup and conduct length extension attacks on MD5 and SHA1.

Forensic: For the problems of this category, beginners mainly need to know the file formats: file signatures, file structures, file headers, file meta information and so on. As discussed in Section 3.2.3, there are a few file types often used in these problems, including PNG, JPEG, PCAP, WAV, AVI, Disk dump, and ZIP. Once beginners are familiar with the formats of these major file types, they will be able to detect hidden information in the files or repair the corrupted files.

5 RELATED WORKS

In recent years, security competitions have been gradually incorporated into the Computer Science education in more and more universities [6,8,9,11,13,14,18]. Despite these efforts and recognized advantages of these new security education approaches, educators and researchers have recognized several issues of security competitions that are particularly challenging to beginners. In [10], six factors were presented to analyze the reasons that security competitions were often very hard to beginners. Among the six factors, three factors were on the design process of competition problems. It was argued that most security problems were designed on heavy technical requirements, some were made harder with artificially added constraints, and many were developed without a proper quality assurance process. For these reasons, the competition problems were not designed for beginners in the first place, and thus led to beginners quickly becoming stuck and giving up.

Additional studies have been conducted to attempt to engage beginners. In [12], a set of small-scoped and hands-on exercises in defense and offense were designed for class use. The goal was to gently introduce beginners to security competitions, rather than simply exposing them to hard problems that they cannot solve. After the exercises, the teacher led an in-class analysis that provided the critical feedback and enabled students to identify the achievements and the areas that require additional practices. In [17], another effort was developed to help beginners. In the study, security problems were divided into several levels. Each level provided a few hints as well as a recommended solution as a last resort. Players could opt to take the hints and the solutions. However, the study did not find a convincing evidence that players were positively benefited from the hints and the solutions. In this

research, we were also concerned on beginners. Our research was focused on helping beginners understand the characteristics of the security competitions and the competition problems so that beginners know what areas and what skills they need to learn in order to participate in the competitions.

6 CONCLUSION

During the course of this paper, we analyzed 160 security competitions taking place during 2014 and 2016 and over 3000 problems in these competitions. The goal of this analysis is to provide a clear picture of the main characteristics of security competitions to beginners. This analysis shows that, with a growing number of security competitions every year, online jeopardy style competitions are the main form of competitions for beginners to participate. There are six dominant categories of problems in these competitions. Each category represents a security area that requires a unique set of knowledge and technical skills, including programming languages, data and file types, and cryptographic algorithms. Therefore, we recommend a few fundamental techniques beginners should study for each category. Currently, we are building a platform with a set of exercises that incorporate the knowledge and skills. We will test the platform in our classes to engage more students in security competitions.

REFERENCES

[1] CTF Time. https://ctftime.org/.

[2] CTF Write-ups. https://github.com/ctfs.

[3] DEF CON Hacking Conference. https://www.defcon.org/.

[4] National Collegiate Cyber Defense Competition. http://www.nationalccdc.org/.

[5] Masooda Bashir, April Lambert, Jian Ming Colin Wee, and Boyi Guo. An Examination of the Vocational and Psychological Characteristics of Cybersecurity Competition Participants. In Proc. of USENIX Summit on Gaming, Games, and Gamification in Security Education, 2015.

[6] Martin Carlisle, Michael Chiaramonte, and David Caswell. Using CTFs for an Undergraduate Cyber Education. In Proc. of USENIX Summit on Gaming, Games, and Gamification in Security Education, August 2015.

[7] Peter Chapman, Jonathan Burket, and David Brumley. PicoCTF: A Game-Based Computer Security Competition for High School Students. In Proc. of USENIX Summit on Gaming, Games, and Gamifi- cation in Security Education, August 2014.

[8] Tom Chothia and Chris Novakovic. An Offline Capture The Flag-Style Virtual Machine and an Assessment of Its Value for Cybersecurity Education. In Proc. USENIX Summit on Gaming, Games, and Gamification in Security Education, 2015.

[9] Tom Chothia and Joeri de Ruiter. Learning From Others' Mistakes: Penetration Testing IoT Devices in the Classroom. In Proc. of USENIX Workshop on Advances in Security Education, 2016.

[10] Kevin Chung and Julian Cohen. Learning Obstacles in the Capture The Flag Model. In Proc. of USENIX Summit on Gaming, Games, and Gamification in Security Education, August 2014.

[11] Adrian Dabrowski, Markus Kammerstetter, Eduard Thamm, Edgar Weippl, and Wolfgang Kastner. Leveraging Competitive Gamification for Sustainable Fun and Profit in Security Education. In Proc. of USENIX Summit on Gaming, Games, and Gamification in Security Education, 2015.

[12] Jelena Mirkovic, Aimee Tabor, Simon Woo, and Portia Pusey. Engaging Novices in Cybersecurity Competitions: A Vision and Lessons Learned at ACM Tapia 2015. In Proc. of USENIX Summit on Gaming, Games, and Gamification in Security Education, 2015.

[13] W. Michael Petullo, Kyle Moses, Ben Klimkowski, Ryan Hand, and Karl Olson. The Use of Cyber- Defense Exercises in Undergraduate Computing Education. In Proc. of USENIX Workshop on Ad- vances in Security Education, 2016.

[14] Aunshul Rege. Multidisciplinary Experiential Learning for Holistic Cybersecurity Education, Research and Evaluation. In Proc. of USENIX Summit on Gaming, Games, and Gamification in Security Education, 2015.

[15] Z. Cliffe Schreuders and Emlyn Butterfield. Gamification for Teaching and Learning Computer Security in Higher Education. In Proc. of USENIX Workshop on Advances in Security Education, 2016.

[16] Giovanni Vigna, Kevin Borgolte, Jacopo Corbetta, Adam Doup´e, Yanick Fratantonio, Luca Invernizzi, Dhilung Kirat, and Yan Shoshitaishvili. Ten Years of iCTF: The Good, The Bad, and The Ugly. In Proc. of USENIX Summit on Gaming, Games, and Gamification in Security Education, 2014.

[17] Jan Vykopal and Milo˘s Bart´ ak. On the Design of Security Games: From Frustrating to Engaging Learning. In Proc. of USENIX Workshop on Advances in Security Education, 2016.

[18] Chuan Yue. Teaching Computer Science With Cybersecurity Education Built-in. In Proc. of USENIX Workshop on Advances in Security Education, 2016.