Capture The Flag

What is a Capture The Flag event?
A capture the flag event is often associated with the physical act of retrieving a flag from an opposition and returning the flag to their home base. This fun game, dating back to the Civil War battlefield, when soldiers would enact the game as a method of improving battle strategy. With the widespread adoption of the internet, individuals' and nations' secrets are now stored online. Similar wargames are held, but now on the digital frontier. These wargames are used to teach and inform others of new and old vulnerabilities in software, with attacks often involving computer forensics, reverse engineering, password cracking and cross-site scripting, amongst other types of challenges.

Popular digital capture the flag events include those offered by SecEDU in Australia, as well as OverTheWire. The most popular events though are often hosted by major corporations like Google and Microsoft, who host these events with large prize pools. The events serve as a great way of gathering lots of intelligent people together to learn new skills from one another, as well as an excellent way of networking with others in the cybersecurity field.

SQL Injection
Structured Query Language, or SQL, is a widely used method of accessing data stored in databases. Lots of programs and services enable a user to input data for a query into these databases. If the users' input has not undergone sufficient sanitization, it would be possible for a malicious user to inject their own executable SQL query into the query sent to the server.

The first public discussions of SQL injections date back to 1988, in the Phrack Magazine. In issue 54 (the December issue), the topic is introduced as an issue with Microsoft SQL Server 6.5. The author, Jeff Forristal, uses examples to demonstrate that the software allows chains of queries without separators, which would enable remote code execution. After being reported to Microsoft, the vulnerability was initially deemed unimportant, with Forristal stating "According to Microsoft, what you're about to read is not a problem, so don't worry about doing anything to stop it."

Over 20 years later, OWASP, the Open Source Foundation for Application Security, rates SQL injection vulnerabilities as one of the most common/severe vulnerabilities. Even now, governments and major corporations face data leaks in the face of SQL injection exploitation.

Let's say that a user is allowed to input a username and password into a login form. That login form uses the SQL query:

```
SELECT * FROM users WHERE username = 'john' AND password = 'doe'
```

This SQL query would be vulnerable to injection if a user were able to add in their own username and password, as is demonstrated below:

```
SELECT * FROM users WHERE username = 'john' AND password='doe' OR password LIKE * AND 'username'='root'
```

Here the malicious user would input the password:

```
doe' OR password LIKE * AND 'username'='root
```

This would cause the SQL server to return all records of users having the username 'root' and a password which has 0 or more characters. Utilizing this, an attacker would have been able to login using the account 'root' and further exploit the system.

XSS

Cross-site scripting, or XSS, is a form of vulnerability where scripts are able to be injected into webpages not owned by the injector. These attacks occur when an attacker sends malicious code (generally in the form of a client-side script) to a different user. XSS enables an attacker to execute these scripts from the perspective of a different user, which ultimately enables access to these vulnerable web applications from a victim's user account.

The first known discussions of XSS go back to late 1999, where a group of Microsoft security engineers from the Microsoft Security Response Centre and the Internet Explorer security team were investigating reports of a variety of XSS attacks, from malicious links being used to get cookies, to the injection of malicious scripts via image HTML tags. Persistent, server-side XSS was also reported, with the payload being stored and re-injected repeatedly. It wasn't until a conference in the fall of 1999, following a presentation of an XSS vulnerability in American Express software, that the Microsoft security engineers began research into these vulnerabilities. Their findings were published in February, 2000 in conjunction with CERT. It was in this report that the term XSS was coined.

Even today, XSS still stands at number 7 on OWASP's top 10 web application security risks. A well-known example of a modern-day XSS is the self-reweeting tweet, which exploited a vulnerability in the Twitter client 'Tweetdeck' to create a tweet which would be retweeted once loaded on the client. With the widespread adoption of frameworks and libraries, XSS is largely tackled by them, with lots of groups benefitting from the work done by a small team.

XSS vulnerabilities can be classed into three categories. Reflective XSS is where data provided by the web client (often in the form of URL parameters) is used by server-side scripts to parse and display data without sanitizing the URL parameters. Persistent/Stored XSS occurs when scripts provided by a malicious entity are stored on the server, and then are displayed to users through the course of their regular browsing. DOM-based XSS is another fully client-side form of XSS, where a malicious user injects JavaScript code to other users' clients.

An example of XSS would be a social networking website which allows for users to add HTML elements to their posts without sanitisation. A user could exploit this DOM-based XSS vulnerability to run scripts on other users' accounts. Possible payloads could involve retrieving another users' session cookies, to running JavaScript-based bitcoin miner on their browser while they browse the website.

Reverse Engineering
Reverse Engineering is a technique used to get code similar to the original source code from an obfuscated piece of software. The technique produces a model of how the original software operates, with better models better reflecting the original product. In a Capture the Flag event, reverse engineering might be used to mimic a given piece of software without input sanitizing or perhaps executing different operations on a dataset.

With regards to the legality of reverse engineering, it is only truly safe to reverse engineer software where the copyright owner has given express permission for their software to be cracked and reversed. International law complicates matters, with DMCA and the 2005 Australia-US Free Trade Agreement ensuring that penalties will be applied if you reverse software illegally.

I am not going to give an example of reverse engineering because it often involves an intricate process of analyzing the assembly of a binary and rebuilding the source code based on the apparent behavior of that assembly. Additionally, I don't want to incur any legal penalties.

Steganography
Steganography is a technique in which data is hidden within another piece of data. This data can later be extracted. Typically, steganography is used to avoid a message being detected. It is often paired with encryption to provide an extra layer of security for the data being hidden.

The first known instances of steganography date back to Ancient Greece around 440BC. The Greek ruler Histaeus used an early form of cryptography which involved shaving the head of a slave, having the secret message tattooed on their head, waiting for their hair to hide the message, and then having that slave sent carrying a different message. The intended recipient would then shave the slave's head in order to get the true message.

A null cipher is another type of steganography. It involves hiding a secret message within an innocuous message. The secret message would be obtained by taking a certain pattern of characters from the innocuous message, for example every third character. An example of a null cipher is:

Fishing freshwater bends and saltwater coasts rewards anyone
feeling stressed. Resourceful anglers usually find masterful
leapers fun and admit swordfish rank overwhelming any day.

When we take every third character, this translates to "Send Lawyers, Guns, and Money."
Reference: http://www.jjtc.com/stegdoc/sec202.html

Steganography has since become much more elaborate. With the introduction of the digital spectrum of technology, messages are now hidden within files, most commonly in images. Additionally, much more inconspicuous techniques of steganography have arisen in which

messages are hidden so well, they are indiscernible to the human eye, and only computers are capable of extracting the data.

A common technique employed in hiding data in images is the manipulation of individual pixels. A message can first be translated to binary, and then the pixels laid on the image such that a 0 corresponds to a black pixel and a 1 corresponds to white. To make things more complex, many use offsets to make it harder to find which pixels actually correlate to the original message.

Cryptography

Secrets have become so engrained in our way of life, that now everybody seems to have them. The trick with secrets is that we need a way of making sure that the secrets aren't easy to uncover. That's where cryptography comes in. Almost every modern company uses encryption and hashing to ensure that is can safely store data without having to worry about leaks. They use a variety of types of encryption to make sure that the secrets which people entrust them remain secret.

The easiest type of encryption would be a substitution cipher. A substitution cipher maps one character to another, with the resulting string supposed to look dissimilar to the original. A simple type of substitution cipher is a Caesar cipher, in which the letters of the alphabet are rotated around an arbitrary amount of characters. The resulting string is easier to decrypt/brute-force as there are only 25 rotations required to achieve all possible permutations, but this is still more complex than having no encryption at all.

Another type is a vigenere cipher. A vigenere cipher takes an arbitrarily long string as a key. It cycles the letters of the key until it matches the length of the plaintext. A table is constructed where each row has the alphabet shifted by 1 more than the previous row. This table is referred to as a Vigenere square. For each letter of the key, you navigate to the row which starts with that letter and find the corresponding letter for the ciphertext. You repeat this process for each character of the plaintext until you have a complete ciphertext.

Similarly, a One-Time Pad (OTP) uses a random key to encrypt data. This random key is able to be used for encryption as well as decryption of the data. The encryption process involves using modular arithmetic, in that the process involves XORing the characters of the key with the ciphertext/plaintext, and then modulo the length of the alphabet.

The earliest known form of cryptography dates back to non-standard hieroglyphs found in the wall of a tomb in Egypt from around 1900BC. Substitution ciphers were used by Hebrew scholars at around 500-600BC. Furthermore, Caesar ciphers and the scytale transposition cipher were used by the ancient Greeks. Since then, cryptography has evolved significantly. With the modern age has come the introduction of computer systems which are capable of encrypting and decryption significantly faster than pen and paper. With this being said, the ability for one to brute-force encryption has become easier too. Therefore, modern encryption is significantly more advanced than ever before to combat this.

References

- Steganography – SEC202. (n.d.). Retrieved from http://www.jjtc.com/stegdoc/sec202.html
- A brief history of encryption. (2016, April 18). Retrieved from https://www.gemalto.com/review/Pages/a-brief-history-of-encryption.aspx
- A History of Cryptography. (n.d.). Retrieved from http://www.cypher.com.au/crypto_history.htm
- Blue, V. (2014, June 12). TweetDeck wasn't actually hacked, and everyone was silly. Retrieved from https://www.zdnet.com/article/tweetdeck-wasnt-actually-hacked-and-everyone-was-silly/
- Capture the flag. (2020, February 9). Retrieved from https://en.wikipedia.org/wiki/Capture_the_flag#Computer_security
- COMP6841 20T1 Wk09 - Reverse Engineering - Clifford Sesel. (2020). Retrieved from https://www.youtube.com/watch?v=CjSuUTvWLm4
- Cox, J. (2015, November 20). The History of SQL Injection, the Hack That Will Never Go Away. Retrieved from https://www.vice.com/en_us/article/aekzez/the-history-of-sql-injection-the-hack-that-will-never-go-away
- Cross Site Scripting (XSS). (n.d.). Retrieved from https://owasp.org/www-community/attacks/xss/
- Cross-site scripting. (2020, March 30). Retrieved from https://en.wikipedia.org/wiki/Cross-site_scripting
- Forristal, J. (n.d.). Phrack Magazine. Retrieved from https://web.archive.org/web/20140319065810/http://www.phrack.com/issues.html?issue=54&id=8#article
- How to Prepare for a Capture the Flag Hacking Competition. (n.d.). Retrieved from https://www.cbtnuggets.com/blog/training/exam-prep/how-to-prepare-for-a-capture-the-flag-hacking-competition
- Oberfelder, R. (2017, August 1). Describing XSS: The story hidden in time. Retrieved from https://medium.com/@ryoberfelder/describing-xss-the-story-hidden-in-time-80c3600ffe81
- One-time pad. (2020, March 29). Retrieved from https://en.wikipedia.org/wiki/One-time_pad
- Siper, A., Farley, R., & Lombardo, C. (2005). The Rise of Steganography. Retrieved from http://csis.pace.edu/~ctappert/srd2005/d1.pdf
- SQL injection. (2020, March 24). Retrieved from https://en.wikipedia.org/wiki/SQL_injection#History
- Vigenère cipher. (2020, March 7). Retrieved from https://en.wikipedia.org/wiki/Vigenère_cipher