Collaborative Discussion 1: UML flowchart- Initial Post

Injection as one of the top ten OWASP vulnerabilities

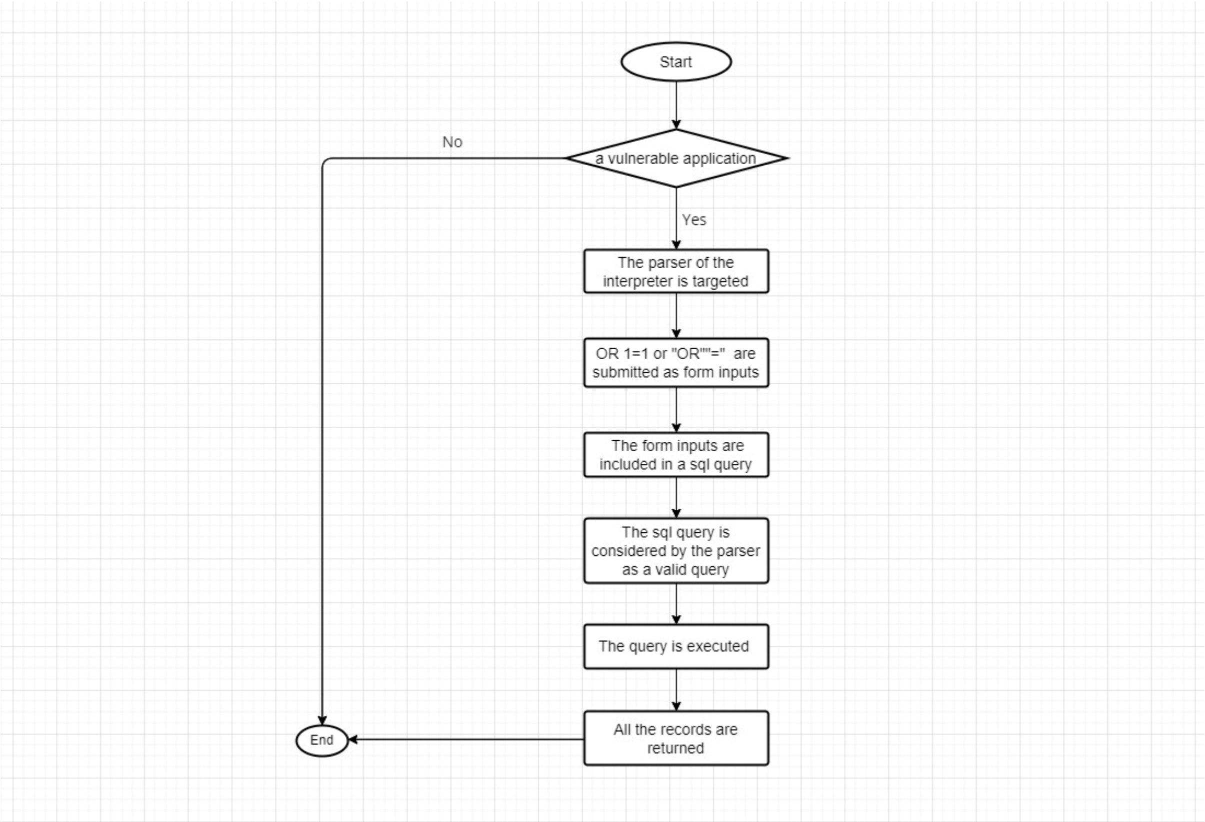
To protect websites from attacks, the Open Web Application Security Project (OWASP) has introduced ten web security vulnerabilities that among them, ‘Injection’ is very prevalent (imperva, 2022). The attacker injects malicious input to an application to change the meaning of the commands being sent to interpreters (Williams, 2022). In fact, attackers target the parsers of the interpreters. So, to do injection attacks or create defence against those attacks, people need a good understanding of the way the parsers work.

There are different types of injection attacks. Among all, SQL injection attacks are considered as one of the most serious vulnerabilities of web applications (Halfond et al., 2006). The attack occurs when attackers insert malicious code into vulnerable applications using different input mechanisms including user input and cookies. In the case of user input, the source of input is form submissions. Attackers submit well-known *OR 1=1* or *"OR ""="* as form inputs. The inputs are included in SQL queries. This trick pushes the database to consider the SQL query as valid one; The reason behind it is that *OR 1=1* or *"OR ""="* always return True. So, the database executes the query and returns all the records of the tables instead of just one (W3schools, 2022).

Protecting applications from injection attacks has not been easy. Traditionally, ‘input validation’ was used to handle malicious data, and everyone believed that it was the most preferred approach (Williams, 2022). They gradually found out that the approach was not a complete solution because it did not treat characters as data but as characters related to the interpreters’ parsers. To solve the problem, ‘escaping’ is used to inform the interpreters that the untrusted data is not going to be executed because SQL server treats special characters that are added to the input as injection attack.

References

- N.D. (2022). OWASP. Available form: https://www.imperva.com/learn/application-security/owasp-top-10/ [Accessed 21 June 2022]
- Williams, J. (2022). Injection Theory. Available from: https://owasp.org/www-community/Injection_Theory#:~:text=Injection%20is%20an%20attacker's%20attempt,instead%20of%20just%20%E2%80%9C101%E2%80%9D [Accessed 21 June 2022]
- Halfond, W., Viegas, J., Orso, A. (2006). A Classification of SQL Injection Attacks and Countermeasures. Available from: https://www.cc.gatech.edu/fac/Alex.Orso/papers/halfond.viegas.orso.ISSSE06.pdf [Accessed 21 June 2022]
- N.D. (2022). SQL Injection. Available from: https://www.w3schools.com/sql/sql_injection.asp [Accessed 28 July 2022]


Start
a vulnerable application
No
Yes
The parser of the interpreter is targeted
OR 1=1 or "OR""=" are submitted as form inputs
The form inputs are included in a sql query
The sql query is considered by the parser as a valid query
The query is executed
All the records are returned
End

Collaborative Discussion 1: UML flowchart- Peer Response

Hi Lukman, thank you for your essay.

I would like to pick up on the part of your comment that injection attacks result in data breaches - neglected by developers and some organizations considering how it effects on individuals!

Data breach is the release of private data or sensitive information into an unsecured environment (Cloudflare, 2022). The data is released accidentally or deliberately. In the case of deliberate release, attackers target big companies because they have databases of individuals' information. There are different ways data breaches can occur including lost/stolen credentials (i.e.: access logins and passwords), social engineering attack (i.e.: trick people to give their information), insider threats (i.e.: release data by people inside a system) and vulnerability exploits which is software related. In the case of vulnerable software, attackers attack the applications by injecting malicious SQL codes to the system and tricking the databases.

The data breaches which occur can impact individuals lives. Using individuals' identity theft, attackers can access to everything including banking information and social security numbers (Kaspersky, 2022). They can do criminal acts under individuals' names. In fact, identity theft can ruin people's credit and even pin them with legal issues. To protect people, all organizations must report personal data breaches to the relevant authority based on the duty introduced by UK General Data Protection Regulation (GDPR) (ICO, 2022).

References

- N.D. (2022). What is a data breach? Available from: https://www.cloudflare.com/en-gb/learning/security/what-is-a-data-breach/ [Accessed 25 June 2022]
- N.D. (2022). How data breaches happen. Available from: https://www.kaspersky.com/resource-center/definitions/data-breach [Accessed 25 June 2022]
- N.D. (2022). Personal data breaches. Available from: https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/personal-data-breaches/#whatisa [Accessed 25 June 2022]

Collaborative Discussion 1: UML flowchart- Summary Post

Today, many companies are involved with the productions of software. So, different aspects of software development have been their major concerns. To develop software, different stages including analysis, UML-based design and implementation should be concentrated on. Every stage really matters because minor mistakes have serious consequences which are not easy to deal with. For example, in design stage, developers may choose a design pattern incorrectly and create software that collapse in normal situations (Phillips, 2018). To create software, developers benefit from several software methodologies including Waterfall and Agile. The traditional Waterfall methodology is useful for simple projects with less change because returning to the early stages of development is difficult and costly. On the contrary, Agile method is beneficial for industry applications with constant changes which need fast completion.

There has always been an issue in creating software which is related to the security of the application. Normally, in software development, the security of the software is considered as a separate layer and added at the end of all stages. Unreliable software is the output of this point of view. To solve the problem, secure software development was emerged where security is embedded from the early stage of development (NCSC, 2022). In fact, it creates secure codes which are resistant to security vulnerabilities. Some common security vulnerabilities include buffer overflow errors, unvalidated input, improper access control, cryptography issues and insecure authentication.

The security vulnerability is even more prominent when Agile methodologies are used. In SCRUM as a popular Agile methodology, sprints are used. To make a secure software, some developers believe that the security should be included in every sprint. But this method may delay the development process. To solve the problem, they introduced Secure SCRUM which mostly benefits from increasing the security awareness of development team (Nguyen et al., 2015).

Python like many programming languages suffers from some security vulnerabilities. To address the vulnerabilities, Python Open Web Application Security Project (OWASP) has been trying to create versions of the language resilient to security attacks (Pillai, 2017: chapter 6). OWASP also referred to other vulnerabilities that among them, injection is very common (imperva, 2022). The attacker injects malicious input to an application to change the meaning of the commands being sent to interpreters (Williams, 2022).

To look closely, we can see that security is such big problem which managing it is a real concern.

References

- Phillips, D. (2018). Python 3 object-oriented programming. 3rd ed. Packt Publishing.
- Pillai, A. B. (2017). Software architecture with Python. 1st ed. Packt Publishing.
- N.D. (2022). Secure development and deployment guidance. Available from: https://www.ncsc.gov.uk/collection/developers-collection/principles/secure-development-is-everyones-concern [Accessed 2 July 2022]
- Nguyen T & Sauter (2015). Software development methods. Available from: https://www.umsl.edu/~sauterv/analysis/F2015/Integrating%20Security%20into%20Agile%20methodologies.html.htm [Accessed 5 July 2022]

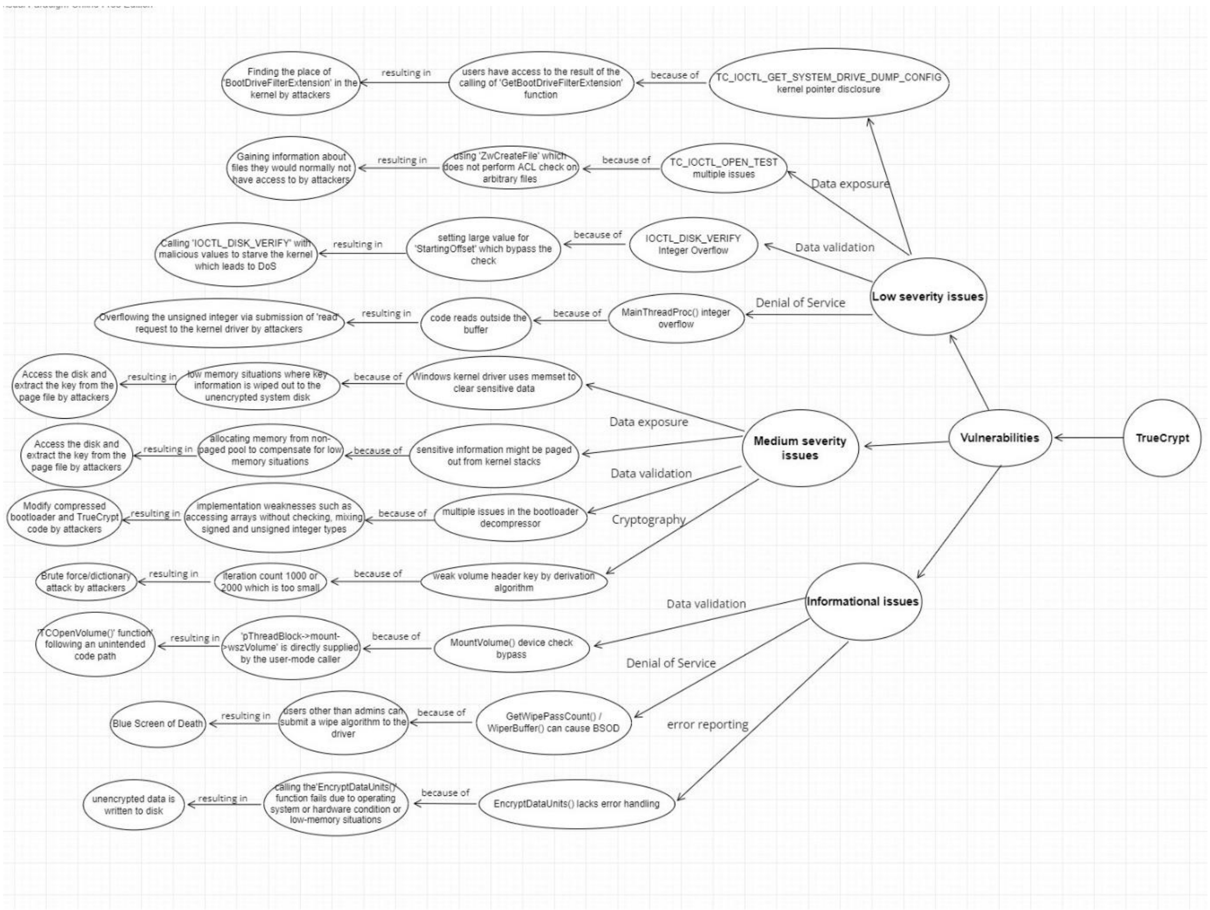
- N.D. (2022). OWASP. Available form: https://www.imperva.com/learn/application-security/owasp-top-10/ [Accessed 21 June 2022]
- Williams, J. (2022). Injection Theory. Available from: https://owasp.org/www-community/Injection_Theory#:~:text=Injection%20is%20an%20attacker's%20attempt,instead%20of%20just%20%E2%80%9C101%E2%80%9D [Accessed 21 June 2022]

Collaborative Discussion 2: Cryptography Case Study- Initial Post

TrueCrypt authors believed that the software is not secure due to unfixed security issues. iSEC engineers audited the software and reviewed bootloader and Windows kernel driver for security issues. After assessing the source code, they found some issues such as inconsistent variable types and deprecated functions. Besides, checking Volume Head integrity showed a weakness which was not a big issue. Similarly, malicious code was not a problem in the software. Therefore, engineers made two high level recommendations- updating Windows build environment and improving code quality (Junestam et al., 2014). Considering above-mentioned subjects, it is concluded that the software is not insecure, and this disproves TrueCrypt authors' assumptions. However, some precautions are needed for using the software; first, avoid low-memory situations which cause data exposure; second, prevent integer overflowing that causes Denial of Service which leads to Blue Screen of Death; finally, using strong volume header key to stop brute-force attacks (Junestam et al., 2014). Taking into consideration those precautions and iSEC recommendations, the software would be a secure and reliable environment. It could be even more reliable and secure in the next editions if continued or turned into an open-source software with a large community to fix bugs.

References

- Junestam, A., Guigo, N. (2022). Open Crypto Audit Project TrueCrypt. Available from: https://opencryptoaudit.org/reports/iSec_Final_Open_Crypto_Audit_Project_TrueCrypt_Security_Assessment.pdf [Accessed 18 August 2022]


TrueCrypt
Vulnerabilities
Low severity issues
Medium severity issues
Informational issues
Data exposure
TC_IOCTL_GET_SYSTEM_DRIVE_DUMP_CONFIG kernel pointer disclosure
because of
users have access to the result of the calling of 'GetBootDriveFilterExtension' function
resulting in
Finding the place of 'BootDriveFilterExtension' in the kernel by attackers
TC_IOCTL_OPEN_TEST multiple issues
because of
using 'ZwCreateFile' which does not perform ACL check on arbitrary files
resulting in
Gaining information about files they would normally not have access to by attackers
Data validation
IOCTL_DISK_VERIFY Integer Overflow
because of
setting large value for 'StartingOffset' which bypass the check
resulting in
Calling 'IOCTL_DISK_VERIFY' with malicious values to starve the kernel which leads to DoS
Denial of Service
MainThreadProc() integer overflow
because of
code reads outside the buffer
resulting in
Overflowing the unsigned integer via submission of 'read' request to the kernel driver by attackers
Data exposure
Windows kernel driver uses memset to clear sensitive data
because of
low memory situations where key information is wiped out to the unencrypted system disk
resulting in
Access the disk and extract the key from the page file by attackers
sensitive information might be paged out from kernel stacks
because of
allocating memory from non-paged pool to compensate for low memory situations
resulting in
Access the disk and extract the key from the page file by attackers
Data validation
multiple issues in the bootloader decompressor
because of
implementation weaknesses such as accessing arrays without checking, mixing signed and unsigned integer types
resulting in
Modify compressed bootloader and TrueCrypt code by attackers
Cryptography
weak volume header key by derivation algorithm
because of
iteration count 1000 or 2000 which is too small
resulting in
Brute force/dictionary attack by attackers
Data validation
MountVolume() device check bypass
because of
'pThreadBlock->mount->wszVolume' is directly supplied by the user-mode caller
resulting in
'TCOpenVolume()' function following an unintended code path
Denial of Service
GetWipePassCount() / WiperBuffer() can cause BSOD
because of
users other than admins can submit a wipe algorithm to the driver
resulting in
Blue Screen of Death
error reporting
EncryptDataUnits() lacks error handling
because of
calling the'EncryptDataUnits()' function fails due to operating system or hardware condition or low-memory situations
resulting in
unencrypted data is written to disk

Collaborative Discussion 2: Cryptography Case Study- Peer Response

Hi Djordje, thank you for your essay.

I would like to pick up on the part of your comment that TrueCrypt is a good disk encryption software and secure storage environment.

An advantage of TrueCrypt is the availability of its audited source code. The audit showed that it had no critical security vulnerabilities and the reason behind it was that the software was relatively secure by default and had no backdoor (StackExchange, 2015). The key challenge of modern software development is writing code which is inherently secure (Pillai, 2017). TrueCrypt as an old-fashioned software could meet the needs of today coding which is remarkable.

TrueCrypt used AES as encryption method which is still in use in various applications and can prevent brute-force attacks (Hougen, 2021).

References

- N.D. (2015). TrueCrypt vs BitLocker. Available from: https://security.stackexchange.com/questions/85149/truecrypt-vs-bitlocker [Accessed 20 August 2022]
- Pillai, A. B. (2017). Software architecture with Python. 1st ed. Packt Publishing.
- Hougen, A. (2021). What is AES Encryption & How does it work? Available from: https://www.cloudwards.net/what-is-aes/ [Accessed 20 August 2022]

Collaborative Discussion 2: Cryptography Case Study- Summary Post

In the software development world, software architecture plays a key role because it represents the design of the entire system. The architecture provides insights about aspects such as scalability, availability, and security of the system. So, every system should have an architecture and the type of the chosen architecture depends on the problem the system is going to solve. Traditionally, to build applications, monolith patterns (top-down) were used which had three layers of client side, server side and data access (Pillai, 2017). The result was large applications which were not easy to understand or change. The advent of microservice patterns solved the problem. Microservices split the applications into independent services which could be scaled and deployed separately. Moreover, services got query from independent local databases.

In modern applications, the focus of the architecture is on security. Apart from software security in architecture, secure coding principles should also be considered. To avoid security crises, different techniques can be applied; Access control is a security technique to associate user roles with certain system privileges (Pillai, 2017). Authentication is another technique which is used to validate the actual identity of the user. Common methods are used in authentication include cryptography and message signing; Using cryptography, the sender and recipient can verify each other’s identities and ensure a safe transfer of information (Encryption Consulting, 2022). To secure the information, they use coded messages for encryption and decryption processes.

To emphasize the importance of security, a case study will be analysed; TrueCrypt is an encryption software which has been discontinued for security reasons. Among problems, a cryptographic vulnerability was recognised by security experts. The problem was related to volume header key derivation algorithm. TrueCrypt uses a very small iteration count, so attackers can easily do password guessing attacks and finally this leads to brute-force attacks (Junestam et al., 2014).

References

- Pillai, A. B. (2017). Software architecture with Python. 1st ed. Packt Publishing.
- N.D. (2022). What Is cryptography in security? Available from: https://www.encryptionconsulting.com/education-center/what-is-cryptography/ [Accessed 29 August 2022]
- Junestam, A., Guigo, N. (2022). Open Crypto Audit Project TrueCrypt. Available from: https://opencryptoaudit.org/reports/iSec_Final_Open_Crypto_Audit_Project_TrueCrypt_Security_Assessment.pdf [Accessed 30 August 2022]