TECHNICAL CONTENT SUMMARY

Hackflix & Skill

securing

HACKFLIX & SKILL

technical content summary

Episode 1 – Buggy's first hack

In the first episode, the participants will meet Buggy. A young, funny (and a bit cocky) hacker who tells us the story of one of his big hacks. Buggy uses an incorrect implementation of an access control mechanism in a web application to get to the administrator's function. This function allows him to load licenses and send them in XML format. Incorrect parser configuration proves vulnerable to an XXE attack, which due to the lack of proper control on outgoing connections, ultimately allows Buggy to steal the organization's internal files.

The episode is based on a kill chain with the following vulnerabilities:

1. Incorrect implementation of access control mechanism.
2. Faulty XML parser configuration allowing XEE attack.
3. Lack of control over outgoing connections.

Issues covered in this episode:

- Access control flaws
    - Access control vulnerabilities - what are they and what do they look like?
    - What is authentication and what is authorization?
    - What are the risks of access control vulnerabilities?
    - Differences between function and resource access control bugs.
    - Examples of the most common access control vulnerabilities.
    - Examples of the most common incorrect fixes.
    - What you should pay attention to in order to correctly implement access control.
- XML vulnerabilities

    - An XML External Entity (XXE) attack - what is it and what does it look like?
    - What is an XML Bomb attack and what does it look like?
    - What are the risks of XXE and XML Bomb attacks?
    - How do attackers exploit incorrect implementations of XML parsers?
    - Default parser configurations are not always properly configured
    - What should a safe XML parser configuration contain?
- Outgoing connections
    - Consequences of not checking outgoing connections.
    - Correct configuration of outgoing connections.
- Introduction to threat modelling
    - What is threat modeling and what are its benefits?
    - Example of basic threat modeling.
    - A set of sample questions to consider during threat modeling.

The episode also includes exercises that help understand the raised issues and enable better knowledge assimilation:

EXERCISE 1: Access control bugs related to functions and resources

An exercise in which the participants - on the basis of, e.g., a description of a situation, sample HTTP requests or http responses that we encountered during the tests - will have to decide whether a given example concerns vulnerability related to a function or to resources.

Purpose of this exercise:

- See a real example of an access control vulnerability.
- Understand how access control errors are exploited and what they are.
- Distinguishing between function and resource access control bugs.

EXERCISE 2: Review of sample payloads used by an attacker to take advantage of incorrect XML parser configuration

An exercise in which the participants will have to choose one of the prepared payloads that allows an attacker to perform a specific action, e.g., download a file, make the server unavailable.

Purpose of this exercise:

- Increased awareness of the consequences of incorrect XML parser configuration.
- Contact with a real example of XXE, XML BOMB.
- Understand how invalid XML parser configurations can be used.

EXERCISE 3: Identifying invalid XML parser configurations

An exercise in which the participants, seeing a sample XML parser configuration, will have to decide whether it is vulnerable or secure.

Purpose of this exercise:

- Realization that not every available configuration is safe with default settings.
- Understanding the key elements that affect security of an XML parser.

- Ability to recognize unsafe XML parser configurations.

EXERCISE 4: Introduction to threat modeling

An exercise in which the participants learn the basics of threat modeling and see what short threat modeling may look like. Then they will try to perform such modeling using prepared sample questions.

Purpose of the exercise:

- To understand the basics, purpose, and how threat modeling works.
- To become familiar with the basic terms.
- To encourage reflection on the designed components within the company.

At the end of the episode, a summary Quiz will allow students to repeat the most important issues raised in the first module.

Episode 2 – Cryptocurrency Baron

In the second episode, Buggy investigates a data leak from a company. Initially, without any access to an application, he notices a mass of redundant information revealed in common places such as HTTP headers, comments, and error pages. Among them, he finds a vulnerable library that allows the administrator to take control of the server. To get an account in the application, he searches the Internet for the activity of the company's employees. It goes to the repository on GitHub, where, by accident, sensitive data allowing access to the application has been placed. Having a user account, Buggy starts fuzzing the application and goes to the log-holding module. There is an old but still working administrator cookie in the logs. From the administrator's account, it uses a component with a known vulnerability and takes control of the server.

The episode is based on a kill chain with the following vulnerabilities:

1. Redundant information.
2. Incorrect data logging.
3. Component with a known vulnerability.

Issues covered in this episode:

- Redundant information
    - What is redundant information in the application?
    - Why is it worth taking care not to reveal excessive information?
    - Examples of where hackers look for information.
    - Examples of places where redundant information is most often revealed.
    - What is important for the correct configuration of responses and HTTP headers?
- Incorrect data logging
    - Security of logs in the admin interface.
    - What are the risks of incorrect monitoring?
    - Examples of the most common monitoring errors.
    - Security basics regarding user sessions.
    - Examples of what is worth monitoring and what is not.
- A component with a known vulnerability
    - How to check the safety of components before using them?
    - What are the risks of using components with known vulnerabilities?

    - Responsible approach to component updates.
- Additionally
    - What is Privilege Escalation?
    - Differences between horizontal and vertical escalation.
    - What is worth paying attention to in the context of the secure implementation of the admin panel?
- Continuation of the introduction to threat modeling
    - Another example of quick threat modeling.
    - A set of sample questions to consider when modeling threats.

The episode also includes exercises that help understand the raised issues and enable better knowledge assimilation:

EXERCISE 1: Horizontal and vertical privilege escalation.

An exercise in which the participant, based on, e.g., a description of the situation, example requests, HTTP responses that we encountered during the tests, will have to decide whether the given example concerns the escalation of horizontal or vertical of user's permissions.
Purpose of the exercise:

- See real examples of privilege escalation and understand the most common privilege escalation vectors.
- Distinguishing between horizontal and vertical escalation.

EXERCISE 2: Sensitive data review.

An exercise in which the participant will have to decide whether a given example should be considered as sensitive data.
Purpose of the exercise:

- Presentation of the most common places where we find redundant information during tests.
- Understanding examples of data that should not be publicly available.

EXERCISE 3: Advising coworkers on error handling

An exercise in which the participant helps the "co-workers" to handle specific security issues related to the topics covered in the module.
Purpose of the exercise:

- Consolidation of appropriate ways to correct the security bugs raised in the module.
- Demonstrating an approach based on openness and help within the team.
- Emphasize the importance and role of the internal security department.

EXERCISE 4: Checking the components

An exercise in which the participant will learn about examples of places containing reliable information and a way to initially recognize the safety of sample components. With the help of the search engine, he will check by himself which CVEs have selected components.
Purpose of the exercise:

- Building the habit of verifying components before use.

- Getting acquainted with places from which it is worth obtaining information.
- Raising awareness of the dangers of using external components.

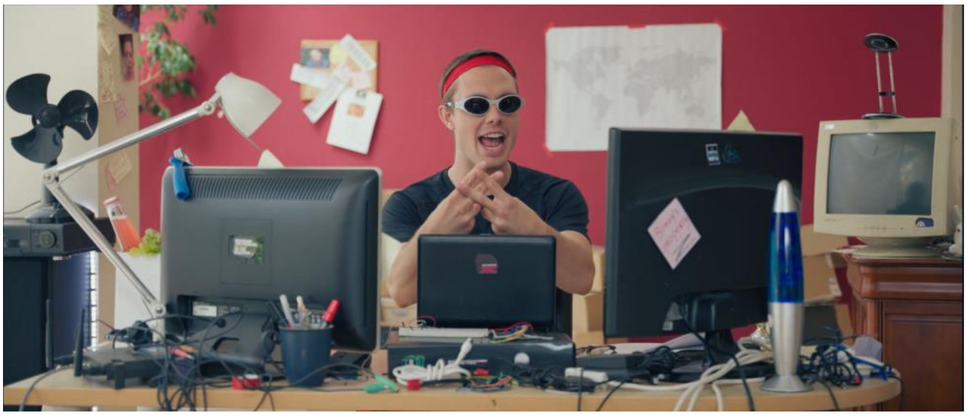
EXERCISE 5: An attempt to independently model threats

An exercise in which the participant will become familiar with the basics of threat modeling and see what a short threat modeling can look like. Then, he will be able to conduct them himself, using the prepared sample questions.
Purpose of the exercise:

- Understand the basics, purpose and how threat modeling works and familiarize yourself with basic terms.
- To encourage reflection on the designed components within the company

At the end of the episode, there will be a Quiz that allows you to revisit the most important issues raised in the second module.

Episode 3 – They hacked my account!



In the third episode, Buggy receives a phone call from his friend Alice. The influencer's account has been hacked, and the broken girl asks for help in recovering it. She reported the problem, but this time it was for nothing, you need to act quickly. Buggy looks for vulnerabilities in the application that will allow you to recover the account. Taking advantage of the lack of proper validation in the name of the uploaded file, Buggy performs a Cross-Site Scripting attack and executes JavaScript code in the context of the hacker's session. Is that enough to regain access to Alice's account?

The episode is based on a kill chain with the following vulnerabilities:

1. Incorrect validation of uploaded files.
2. Cross-Site Scripting.
3. No additional authorization for key operations.

Issues covered in this episode:

- Cross-Site Scripting (XSS)
    - What are the Cross-Site Scripting (XSS) vulnerabilities?
    - What are the risks of XSS?
    - Differences between Reflected and Stored XSS.
    - Examples of different places where XSS can occur.
    - Examples of various payloads that can be used to carry out an attack.
    - How to protect yourself against them?
- Validation of uploaded files
    - Ways in which the attacker can take advantage of the lack of proper validation of uploaded files.
    - The risk posed by the lack of proper validation of the uploaded files.
    - How should the uploaded files be validated?
- Authorization of key operations
    - Why should some operations require additional authentication?

    - For which operations is it worth requiring additional authentication?
    - What to pay attention to when implementing the authorization of key operations?
- Additionally
    - Password reset and the risks associated with it.
    - What to be careful so that the account is not taken over?
    - What to do when the account is taken over?
- Continuation of the introduction to threat modeling
    - Another example of brief threat modeling.
    - A set of sample questions to consider when modeling threats.

The episode also includes exercises that help understand the raised issues and enable better knowledge assimilation:

EXERCISE 1: XSS vulnerabilities.

An exercise in which the participant, on the basis of, e.g., a description of the situation, example HTTP requests and payloads contained in them, will have to answer questions related to the XSS vulnerability.

Purpose of the exercise:

- See a real XSS example.
- Understanding how XSS attacks are carried out and what they are all about.
- Learning about the different places in the application that may be affected.
- Getting acquainted with examples of the effects of using a vulnerability.

EXERCISE 2: How not to fix an XSS?

An exercise in which the participant will learn about examples of erroneous vulnerability remediations and see how they are circumvented.

Purpose of the exercise:

- Marking incorrect approach to vulnerability remediation
- Demonstrating ways to bypass validation by the attacker.
- Make developers aware that an attacker can launch an attack in a number of ways.

EXERCISE 3: Reflected and Stored XSS

An exercise in which the participant, based on the description and HTTP requests, will have to decide whether the example is for a Reflected or Stored XSS error.

Purpose of the exercise:

- Understand the difference between Reflected and Stored XSS.
- Understanding where the vulnerability may occur in the application.
- Understand what effects vulnerability can have.

EXERCISE 4: Validation of uploaded files

An exercise in which the participant will learn about typical errors in the functionalities responsible for uploading files.

Purpose of the exercise:

- Understanding how an attacker can take advantage of the lack of proper validation of uploaded files.
- Being aware of the various components of the file and HTTP requests uploading the file that may allow an attack on applications.
- Learning the methods of proper validation of uploaded files.

EXERCISE 5: Authorization of key operations

Exercise in which the participant will get acquainted with the authorization of key operations. Based on the description, he will have to refer to the presented situation.

Purpose of the exercise:

- Understanding why some operations require additional authorization.
- Understanding the example operations to which the additional component should be implemented.
- Understanding how the lack of additional authorization for specific operations can affect the security of the system.

EXERCISE 6: An attempt to independently model threats

An exercise in which the participant will get acquainted with the basics of threat modeling and see how a short threat modeling can be performed. Then, he will be able to conduct them himself, using the prepared sample questions.

Purpose of the exercise:

- Understand the basics, purpose and how threat modeling works.
- Familiarization with basic terms.
- To encourage reflection on the designed components within the company.

At the end of the episode, there will be a Quiz that allows you to revisit the most important issues raised in the third module.

Episode 4 – Good old friends

History may not repeat itself, but it rhymes well. In the fourth episode, Buggy relives the situation from 10 years ago. While looking for a renovation team for his apartment, he comes across a friend whom he once helped secure an application for his first business. Even then, Buggy was such a smart hacker that right after starting the tests, he detected a SQL Injection vulnerability in the login form. The vulnerability allowed him to extract information encrypted with a homemade cryptographic algorithm. Currently, the application has been updated. In addition to using secure cryptographic libraries and a brand-new NoSQL database, some additional changes have been made to keep up with the times. The friend, being sure of this, offered to pay for the renovation if Buggy somehow got into the administrator's account.

The episode is based on a kill chain with the following vulnerabilities:

1. NoSQL Injection.
2. Data wrongly stored in the database.
3. Incorrect use of cryptographic functions.

Issues covered in this episode:

- SQL Injection
    - What is a SQL Injection (SQLi) vulnerability?
    - What are the risks of SQLi?
    - Examples of various payloads that can be used to carry out an attack.
    - Identification of SQLi attack types (e.g., time-based, error-based).
    - How to protect yourself against them?
- Cryptography
    - Examples of common errors when using cryptographic functions.
    - Dangers of creating your own cryptography.
    - Examples of security measures against common errors.
    - The difference between encryption, hashing and encoding.
    - The difference between symmetric and asymmetric encryption.
    - Cryptographic key management:
        - storage,
        - rotation.
- NoSQL Injection & New Database Technologies

  - What is the vulnerability of the NoSQL Injection (NoSQLi) type.
  - What are the risks of NoSQLi?
  - Examples of various payloads that can be used to carry out an attack.
  - How to protect yourself against them?
  - Examples of SQLi and NoSQLi attacks.
- Additionally
  - How to keep confidential information in databases.
  - Example of automated SQLi exploitation.
  - Automation of SQLi vulnerability detection in code through source code analysis tools.
- Continuation of the introduction to threat modeling
  - Another example of brief threat modeling.
  - A set of sample questions to consider when modeling threats.

The episode also includes exercises that help understand the raised issues and enable better knowledge assimilation:

EXERCISE 1: Permission level and resources in the database

Exercise in which the participant, on the basis of, e.g., a short description of the situation, database schema, will have to decide who should be authorized to a given table.

Purpose of the exercise:

- Familiarization with the "Principle of least privilege" rule.
- Realizing what may be possible when assigning too broad permissions to the application user.
- To encourage a rethink of the current approach to permissions in the databases used.

EXERCISE 2: String concatenation vs Prepared Statements & Stored Procedures

An exercise presenting methods of submitting SQL queries in popular languages in a safe and unsafe manner. The participant's goal is to choose the correct implementations based on the presented code fragments.

Purpose of the exercise:

- Familiarization with safe methods of creating SQL queries in popular languages.
- Showing from the code side how dangerous queries look like.
- Understanding where the SQLi vulnerability comes from.

EXERCISE 3: Examples of using ORM, NoSQL and GraphQL

Exercise presenting both safe and dangerous approaches to communication with the database using technologies that are gaining popularity (NoSQL, GraphQL). The participant aims to choose the correct approaches based on the presented code fragments.

Purpose of the exercise:

- Familiarization with safe methods of creating queries using technology - NoSQL / GraphQL.
- Showing from the code side how dangerous queries look like.

- Understanding where the vulnerability comes from in the technologies covered.

EXERCISE 4: Examples of SQLi payloads

An exercise in which the participant, on the basis of, e.g., a description of the situation, example HTTP requests and payloads contained in them, will have to answer questions related to the SQLi type vulnerability.

Purpose of the exercise:

- See an actual SQLi attack.
- Understand how SQLi attacks are performed and what they are all about.
- Learning about examples of places in the application that may be affected by the vulnerability.
- Realization that newer technologies are also susceptible to classic types of attacks.
- Getting acquainted with examples of the effects of using a vulnerability.

EXERCISE 5: Differentiation of types of SQLi attacks

An exercise in which the participant, based on, e.g., a description of the situation or screenshots, aims to recognize what type of SQLi attack was performed.

Purpose of the exercise:

- Getting to know the types of SQLi attacks.
- Understanding how redundant messages (response time, different response code) can be used to extract information from the database.
- Getting to know the methods of securing the application against various types of attacks.

EXERCISE 6: Encryption recognition against reversible forms

An exercise in which the participant, based on, e.g., a description of the situation, example HTTP requests, is aimed at recognizing whether the highlighted fragment is transferred and / or stored in a secure (encrypted) manner or not (readable due to a weak hash function or coding).

Purpose of the exercise:

- Make the participant aware of the difference between encryption, hashing and encoding.
- Familiarize the participant with the dangers of an incorrect choice.

EXERCISE 7: Identifying types of encryption

The exercise by showing different types of situations - e.g., through descriptions, HTTP requests, network packets - checks whether the participant is able to recognize the type of encryption used.

Purpose of the exercise:

- The participant is able to recognize the types of encryption and usage.
- Getting to know the advantages and disadvantages of certain types of encryption.

EXERCISE 8: An attempt to independently model threats

An exercise in which the participant will get acquainted with the basics of threat modeling and see how a short threat modeling can be performed. Then, he will be able to conduct them himself, using the prepared sample questions.

Purpose of the exercise:

- Understand the basics, purpose and how threat modeling works.
- Familiarization with basic terms.
- To encourage reflection on the designed components within the company.

At the end of the episode, there will be a Quiz that allows you to revisit the most important issues raised in the third module.

Episode 5– Remotely, the way I like it

In the fifth episode, the problem concerns the Fintech organization - there are suspicions that it is doing serious financial scams, but this is not so easy to prove. It was decided that the best solution would be the accelerated recruitment of IT security specialists who would be able to find flaws in the system and confirm suspicions of fraud. The challenge is accepted, Buggy starts from the home page, which after a while goes to deserialization. He manages to use it to execute the code, but apart from a few simple operations, it doesn't give him much in terms of finding the evidence. The machine has extra security on the network layer, and the Buggy can't get through them. Without losing hope, he comes up with an idea - what if it is cloud architecture? Bingo! The metadata of the server responds with the access key to the account. The role obtained has extensive powers that allow you to find other keys that already have access to the buckets (storage), where all documents with evidence of committed crimes are located.

The episode is based on a kill chain with the following vulnerabilities:

1. Dangerous Deserialization.
2. Remote Code Execution (RCE).
3. Incorrect configuration of the cloud infrastructure.

Issues covered in this episode:

- Dangerous Deserialization
  - What is the vulnerability in the deserialization process?
  - What are the risks of dangerous deserialization?
  - Examples of various payloads that can be used to carry out an attack.
  - How to protect yourself against them?
- Remote Code Execution
  - What is the Remote Code Execution (RCE) vulnerability?
  - What are the risks of RCE?
  - Practical examples of using the RCE error.
  - How to protect yourself against RCE?
  - Difference between RCE and Command Injection (OS Injection).
- Cloud architecture

  - Examples of common errors in cloud infrastructures.
  - What are the risks of vulnerabilities in cloud infrastructure?
  - An example of the escalation of rights in the cloud infrastructure.
- Additionally
  - Securing the server (Hardening).
- Continuation of the introduction to threat modeling
  - Another example of brief threat modeling.
  - A set of sample questions to consider when modeling threats.

The episode also includes exercises that help understand the raised issues and enable better knowledge assimilation:

EXERCISE 1: Secret or not?

An exercise in which the participant has to decide whether a given information can be stored in text form or whether security mechanisms (e.g., encryption) should be used.

Purpose of the exercise:

- Draw the participant's attention to the way secrets are kept.
- Understanding the risks of unsafe keeping of secrets.

EXERCISE 2: Which line contains the vulnerability?

An exercise in which the participant, based on a code fragment, has to choose the line that is responsible for the RCE vulnerability.

Purpose of the exercise:

- Showing what leads to a RCE vulnerability from the code side.
- Familiarization with methods of protection against RCE attacks.
- Make the participant aware of the need to conduct a code review.

EXERCISE 3: Examples of RCE payloads

An exercise in which the participant, on the basis of e.g. a description of the situation, example HTTP requests and payloads contained in them, will have to answer questions related to the RCE vulnerability.

Purpose of the exercise:

- Seeing a real RCE attack.
- Understanding how RCE attacks are carried out and what they are all about.
- Learning about examples of places in the application that may be affected by the vulnerability.
- Getting acquainted with examples of the effects of using a vulnerability.

EXERCISE 4: Safe or not?

An exercise in which the participant, on the basis of a code fragment, is to decide whether it is protected against vulnerabilities related to deserialization.

Purpose of the exercise:

- Understanding how deserialization attacks work.
- Getting to know examples of places in the code that may be affected by this vulnerability.
- Familiarization with methods of securing against deserialization attacks.

EXERCISE 5: Basics of strengthening security

An exercise in which the participant, based on, e.g., a description of the situation, example HTTP requests, configuration, is to assess which approach is considered safer.

Purpose of the exercise:

- To encourage a rethink of the current approach to securing servers.
- Making the participant aware of the existing security mechanisms.
- Familiarization with good security practices.

EXERCISE 6: An attempt to independently model threats

An exercise in which the participant will get acquainted with the basics of threat modeling and see how a short threat modeling can be performed. Then, he will be able to conduct them himself, using the prepared sample questions.

Purpose of the exercise:

- Understand the basics, purpose and how threat modeling works.
- Familiarization with basic terms.
- To encourage reflection on the designed components within the company.

At the end of the episode, there will be a Quiz that allows you to revisit the most important issues raised in the third module.

securing

contact: info@securing.pl