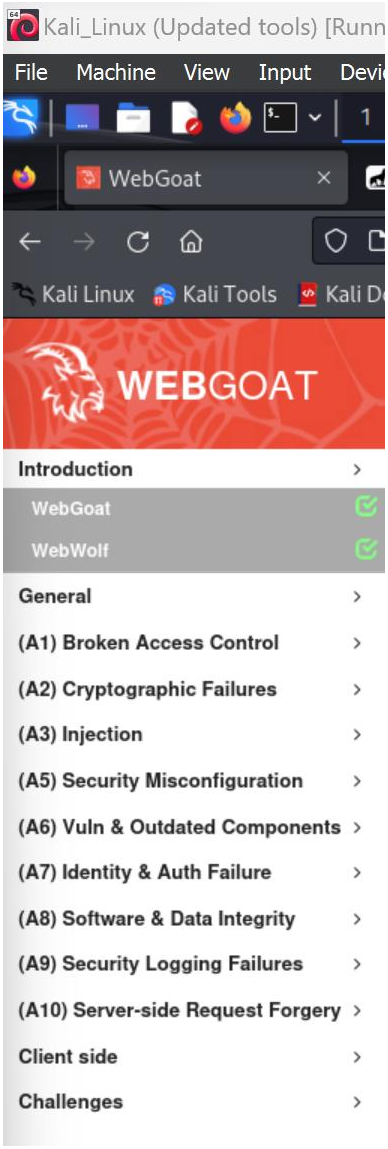
WebGoat – Exercise

By Sebastian Medina

Contents:



1.1 Web Goat

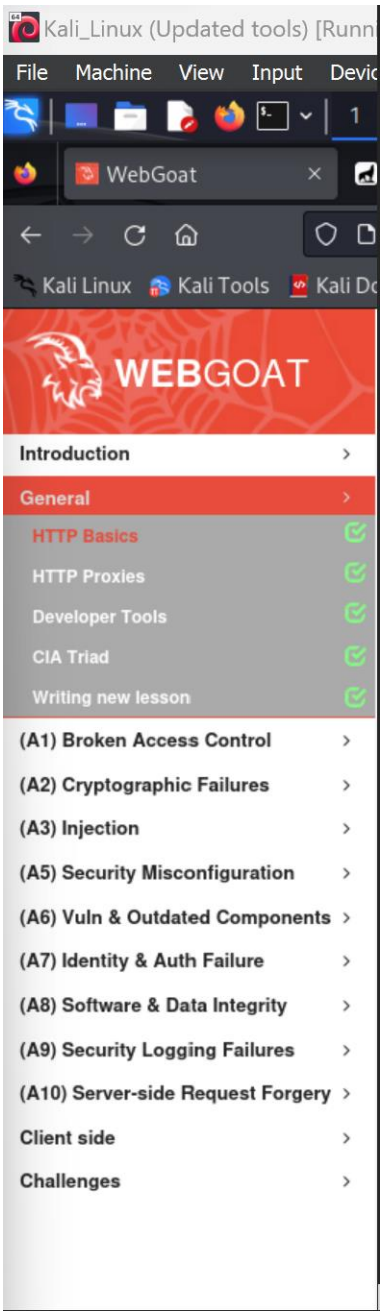
Summary:

For this module it talked about what Webgoat is and what can you do in Webgoat. It also talked about how Webgoat can be used to learn about security vulnerabilities and more.

1.2 Web Wolf

Summary:

Within this module it talked about how WebWolf is supposed to play the role of an attacker to stop any confusion. The module also talked about how WebWolf also has a mailbox client that can contain the e-mail sent during a lesson. Lastly you can use Webwolf as your landing page to harvest cookies and more.



2.1 HTTP Basics

Summary:

In this section we went over the basic of understanding how to transfer data between the browser and the web application works and how you can trap request and response. I also learned how HTTP works and how HTTP has three parts the request or response line, a header and the entity body. In this module it went over some examples as well on how just entering your name into a answer bar triggers a request. I also learned the differences between a Get Request and a Post Request. A GET request can have url parameters that are available in the web access logs. The Post Request are user supplied data and is not part of the contained POST URL.

2.2 HTTP Proxies

Summary:

Within this section I learned about proxy and how it is some forwarder application that connects to your HTTP client to backend resources. This module also taught me how to set up HTTP Proxy Setup with OWASP ZAP however they said if you want to use anything else like burp you can. I ended up using burp for most of the sections. This also taught me how to filter requests in the history panel. However, I did learn that with me using burp I did have to manually configure Burp to work with WebGoat.

2.3 Developer Tools

Summary:

With in this part of WebGoat we talked about Google Chrome Developer Tools and how to complete some of the assignment you will have to look at the JavaScript source code or run

the JavaScript. I challenge I came across is that I was using FireFox instead of Google Chrome, so everything was in a different location but still has the same tools. This section also talked about how to navigate the inspect element and get though the HTML source or the CSS source. It also talks about the different types of tabs that they hav when you use the investigate element. Lastly it had you input some certain flags that they had you find to complete the assighment.

2.4 CIA Triad

Summary:

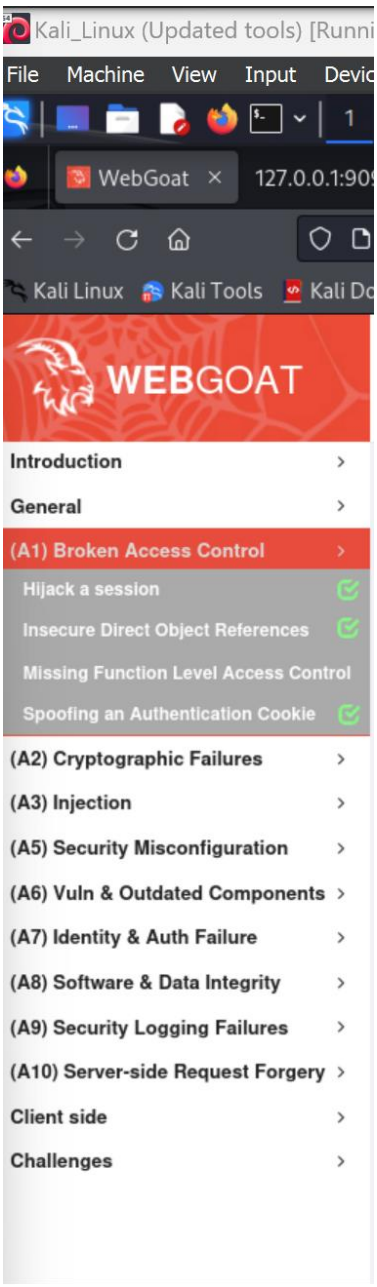
CIA stands for confidentiality, integrity and availability and the CIA Triad is a model for information security. These three components are the most crucial security components and should guarantee any security system for you. The CIA triad is a baseline standard for evaluating implementing security for any organization. Now let's go over the three components in detail. Confidentiality is the property that information should not be made available to unauthorized individuals. Integrity is the property of accuracy and completeness. Lastly there is Availability, and this is the property of being accessible and usable whenever somebody with the right authority calls upon it.

2.5 Writing new lesson

Summary:

I found this section interesting because it taught me how to add a new lesson into WebGoat which is cool. You can completely add a new lesson with only four steps. The first step is writing content explain the vulnerability that is written in AsciiDoc which make is very easy to write content. The second step is adding in the new lesson class, and you do this by defining a lesson class in java. In the third step you write glue html page which allows you to use multiple

assignments quickly and efficiently. Final the last step is adding an assignment to your lesson, this is where you put everything all together. A user starts a lesson and within the lesson you can have as many assignments as you like and that how you create your own lessons.



3.1 Broken Access Control

Summary:

This is the first real lesson that I started in WebGoat and I had to gain access to an authenticated session belonging to somebody else. What I found interesting in the section was that most developers who develop their own session IDs most often then not forget to incorporate the randomness necessary for security. If your session is not complex and/or random, your session is highly susceptible to brute force attacks.

3.2 Insecure Direct Object References

Summary:

To understand this module, you must first understand what Direct Object References are. Direct Object References is when an application uses client- Provided input to access data and objects. Now let's talk about what makes them Insecure, they are considered insecure when reference is not properly handled and allows authorization bypasses. Direct Object Reference are also considered insecure when they disclose private data that can be used to perform tasks or access data that should not be accessed. In this section we also talked about access control and APIs. APIs rely on a obscurity, like static keys or lack of imagination on the user. A good way to secure this is to have a secure token to a certain cryptographic state.

3.3 Missing Function Level Access Control

Summary:

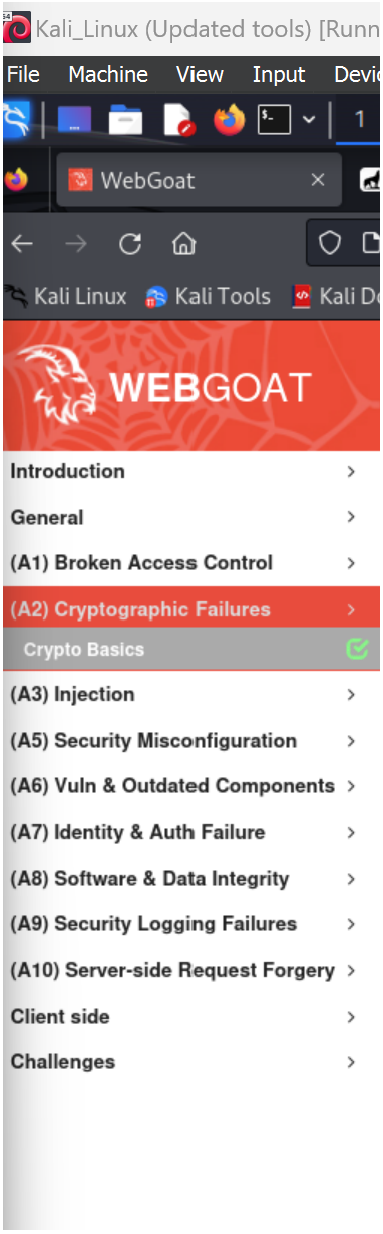
In this module I learned that Access Control can be tricky to maintain but is important to maintain it to have it work properly. I was not able to complete this section, so I did not grasp the whole concept but here is what I understood from it. Some things that can happen if not

maintained correctly are that HTML, CSS, or JavaScript can be used to hide links that users do not see or have access to them. Another problem, however much rarer, is SQL injections. A great description that I read doing this section is to think of Access Control exposes functionality.

3.4 Spoofing an Authentication Cookie

Summary:

The concept of this module is to see how spoofing works and how cookie being stored on the client side means it can be stolen though vulnerabilities on the client side. However, the cookies can be guessed if the algorithm if the code to generate the cookies is obtained from the server-side witch could be worse. It is very important for security that the cookie generation code is protected and not able to be guessed easily. Some good ways to boost security for authentication cookies are adding session expiration and regular rotation.

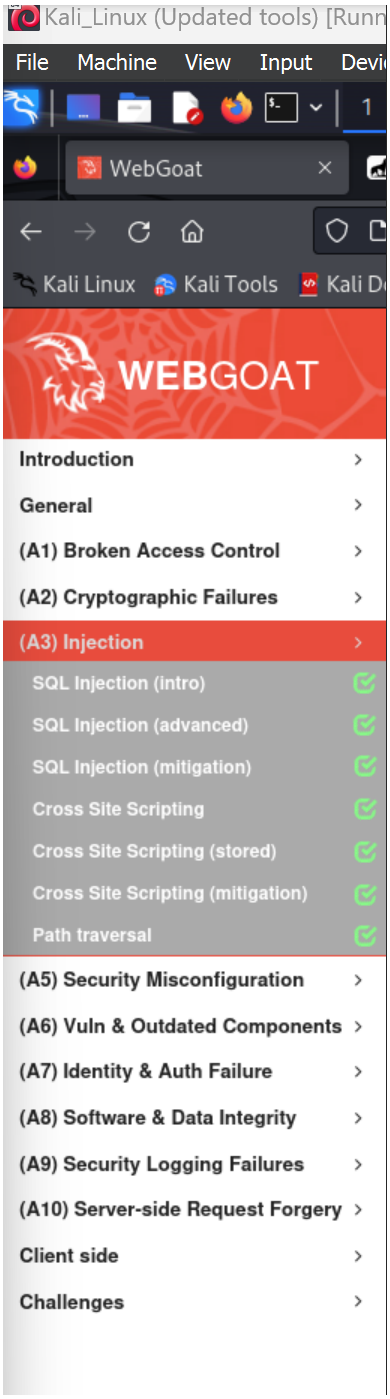


4.1Crypto Basics

Summary:

I did enjoy learning from this module because I learned multiple different types of cryptography techniques are used in web applications. One method is Encoding and even though it is not really cryptography it can still be used around cryptographic functions. A popular

encoding is Base64 that transforms all kinds of bytes to a specific range of bytes. Another popular cryptography is plain hashing that is mostly used to detect to see if the original data has been changed at all. This works because even if only one byte is changed in the data the hash will look different as well and we would be able to tell. Then it brings us to Symmetric and Asymmetric encryption . Symmetric is were there is a shared secret key that is used for both encrypting and decrypting the data, while asymmetric you have a private and public key and only one can be used to encrypt data while the other is used to decrypt the data.



5.1 SQL Injection (intro)

Summary:

In this segment you learn about SQL (Structured Query Language and how they can be manipulated to do certain task that developer original did not intent on performing. SQL is a standardized programming language that manages relational databases and completes multiple tasks on the data within them. One way of manipulating SQL is though DML (Data Manipulation Language) which can request, add, delete, or modify any records within the data. Another way to manipulate the data is to use DCL (Data Control Language) to grant or remove user privileges on database objects. SQL injection is one of the most common web hacking techniques because it can seriously impact the integrity and security of data.

5.2 SQL INJECTION (advanced)

Summary:

This segment is not much different than the previous one, in this segment we talk about the more advanced topics of SQL injection. For example, we talk about the Union operator and how it is used to combine any selected statements together. However, while using Union there are some rules to keep in mind; the columns selected in each statement must be the same, and the datatype of the first column in the first statement must be a match in the rest of the columns. Something interesting that I did not know about is blind SQL injection, were it ask the database true or false questions to find out how the query is working and because of it not having anything displayed makes this SQL injection to exploit.

5.3 SQL Injection (mitigation)

Summary:

In this module we talked about mitigating SQL injection and how to stop or prevent any attacks like Immutable queries. Immutable queries are one of the best defenses against SQL

injection, because they do not have any data to interpret or treat data as a single entity that is assigned to a column. Lastly if you want to mitigate any attacks you also need to provide a sorting column make sure you add a whitelist in order to validate the value of the order by statement and make sure it is limited to something like a first name.

5.4 Cross Site Scripting

In this section you learn how Cross site scripting is used to perform task that the developer did not intent on happing. I form of cross site scripting is XSS is a vulnerability that combines the allowance of script tags as input that renders into a browser without any kind of sanitization. XSS is the most prevalent and pernicious web application security issue because if the data is not protected sensitive data can be stolen and used by somebody else. Ther are multiple types of XSS like reflected, Dom-based, or stored. Reflected is where malicious content from a user is displayed to that user in their web browser. A DOM- based XSS is where attackers use the client's side to insert malicious content from a user request to write HTML to its page. Lastly there is a Stored XSS where malicious content is put onto a server anywhere and later displayed on the users' web browser.

5.5 Cross Site Scripting (stored)

Summary:

In this lesson you learned more about the Cross-site scripting attack: Stored XSS. Like mentioned above the malicious script is inserted into a server database, or a message board and much more. The Stored XSS stelas information like session id and gives it to the attackers.

5.6 Cross Site Scripting (mitigation)

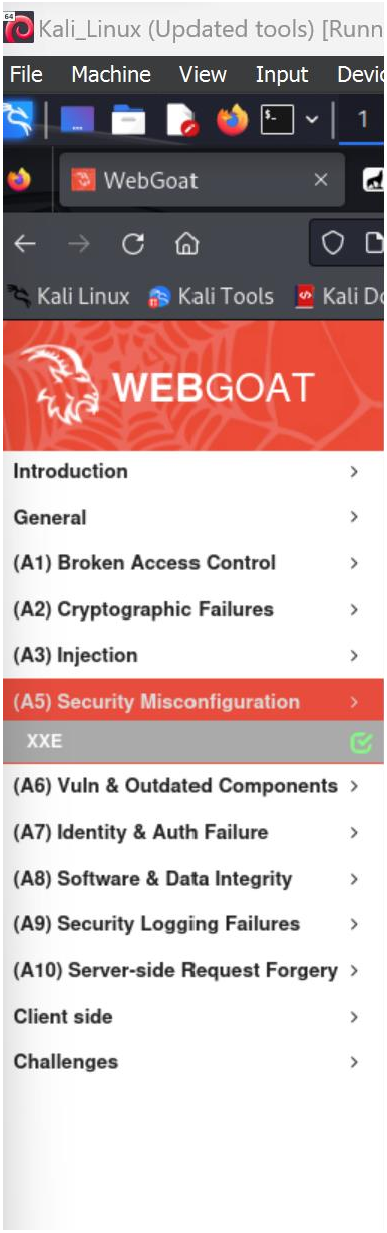
Summary:

Now after learning about cross-site scripting attacks, and how they work, it is time to learn how to defend against them. The main reason to defend against this attack is obvious and that is that somebody else's code running within your users is not good. The basic defense against XSS is to output encoding for any untrusted input for the screen. As of now this is the best way to defend against an XSS attack. However, with these attacks becoming more sophisticated we might have to find another good solution. Another technique to use to prevent attacks is escaping and this is where you convert key characters of data to prevent it from being interpreted in a dangerous context.

5.7 Path traversal

Summary:

Before we talk about path traversal you must know what path traversal is. Path traversal is a vulnerability where an attacker can access or store files and directories away from the application location. Another vulnerability that we covered in this section is Zip Slip. A Zip Slip is when it uses path traversal which is used while extracting files. With a zip slip attack you can even overwrite commands like ls so that every time that certain command is executed it executes some malicious action.

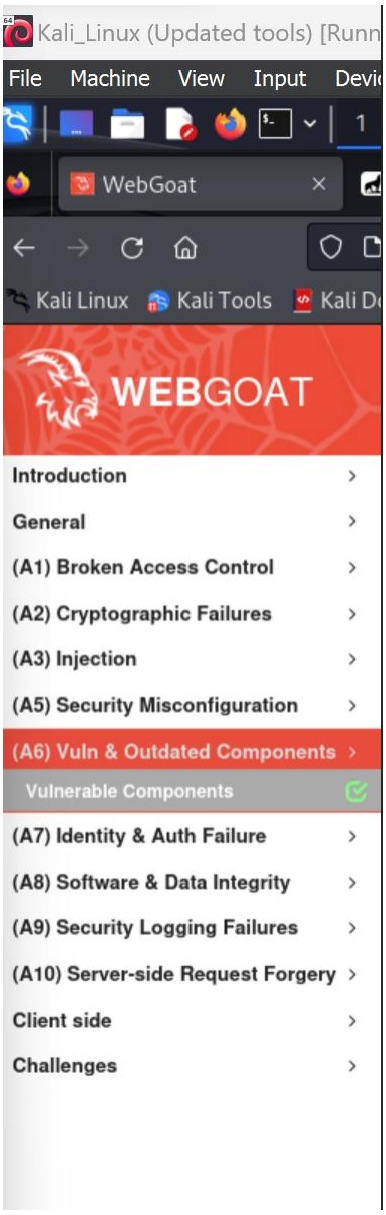


6.1 XXE

Summary:

Within this lesson you learned about how to perform an XML External Entity attack and how to protect yourself from it. So, to understand anything about this topic you have to understand what an XML entity is. An XML Entity makes tags be replaced by content when the

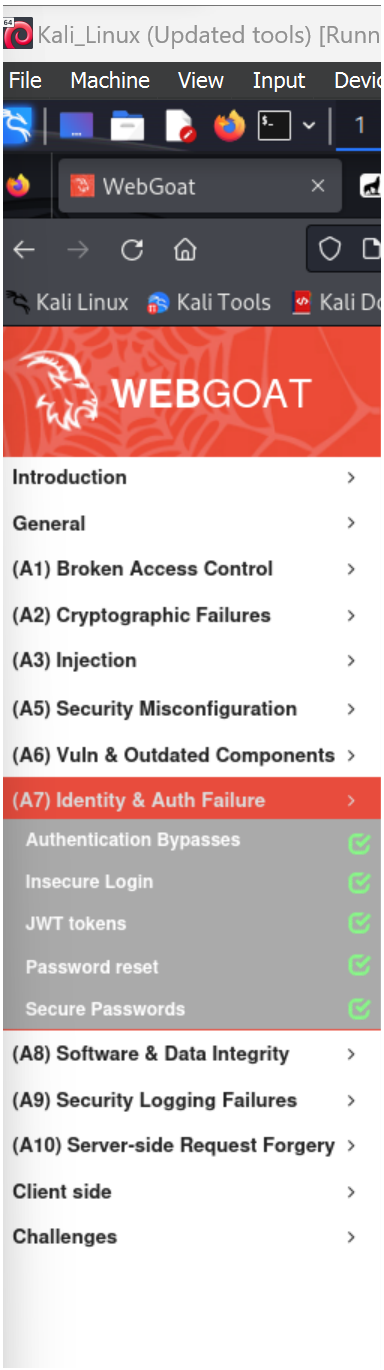
XML Doc Is parsed. Now a XXE injection attack against an application that parses XML input. This attack most of the time happens when the XML input containing a reference to an external entity is processed by a weakly configured XML parser. To protect yourself against these types of attacks you must make sure to validate the input that is being received from an untrusted client.



7.1 Vulnerable Components

Summary:

The concept of this lesson is to inform that the way that we build software has changed. This is because the open-source community is maturing, and open-source software is becoming prolific without having to worry about the provenance of libraries in applications. An interesting fact is that WebGoat uses almost 200 Java and Java Script libraries. Many people think that exploits are just within the code, however this is not the case by any means. Now moving over to License information overload it is important to realize that it is difficult to determine the scop of a license as well as that projects have a license discrepancies.



8.1 Authentication Bypasses

Summary:

An authentication Bypasses happens in many ways but usually takes the advantage of some flaw in the configuration. One of the most common forms to see a authentication bypass is

a hidden input were a reliance on a hidden input within the webpage. Sometimes if it is a less experiences attacker and they do not know the correct value of a parameter, they might remove it from the submission altogether to see what will happen.

8.2 Insecure Login

Summary:

This lesson was simple because we just went over how encryption is an essential tool for secured communication and how it should always be there when talking about sensitive data. In this lab we were able to intercept and read unencrypted requests.

8.3 JWT tokens

Summary:

Within this module it teaches how to use JSON web tokens to authenticate but also what are some common vulnerabilities that you need to be aware of. A JSON web token is a compact way to represent information between two parties. It consists of a signature, a header and a payload. There are multiple ways that JSON web tokens can be attacked like Unauthorized claims, tampering claims and Replay attack. An unauthorized claim is a malicious user who might try to add unauthorized claims to the token to gain access to certain information that they are not authorized to look at. A tampering claim is an attacker who has tried to modify the values of existing claims within the token to manipulate their own identity. Lastly there is a replay attack, and this is when an attacker might try and reuse a valid token from a previous session to gain access by pretending to be the original users.

8.4 Password reset

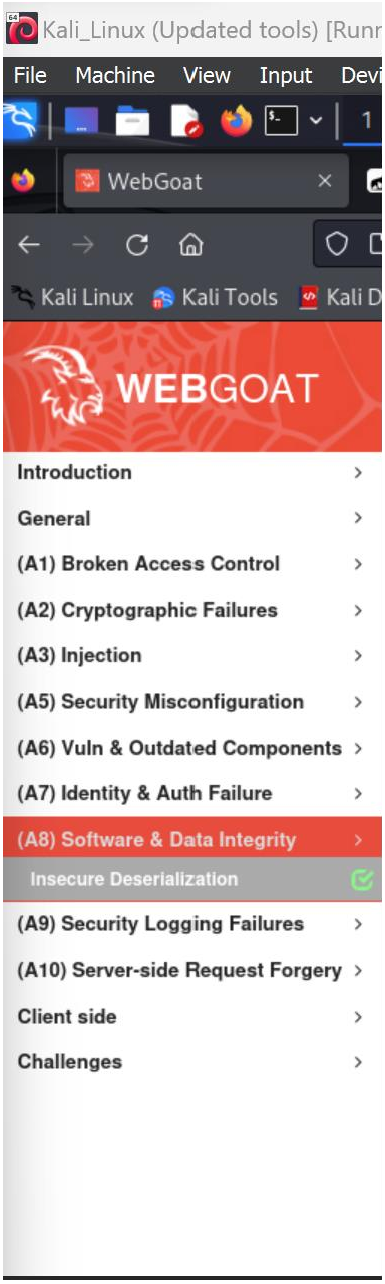
Summary:

In this lesson I learned about password functionality which most often than not is overlooked which can cause multiple vulnerability and logic flaws. Everybody has used a password reset function on some sort of website before and did not think that it could go wrong. For example, there has been an issue about password resets with security question. Most of the time the list of questions contains a fixed number of questions that most often then does not have a limited set of answers. Most of the time this question can be guessed easily with very little information about the users because of how limited the questions are. The best way to prevent this is to make sure that all the question you have to input an answer making it a lot harder to guess.

8.5 Secure Passwords

Summary:

Now one of the most important parts of security for about everything is making sure that you know how to create and secure strong passwords. The NIST (National Institute of Standards and Technology) is a non-regulatory federal agency that soul goal is to advance science, standards and technology that enhance security and quality of life. NIST provide the (SP) 800-serious that have guidelines for security and making a password and is one of the best practices across the security industry. Now to ask your self if your passwords are secure you can check this by making sure that not two or more passwords are the same for different accounts, you use passphrases or a password manger and if you want to make it more secure you can use two-factor authentication.

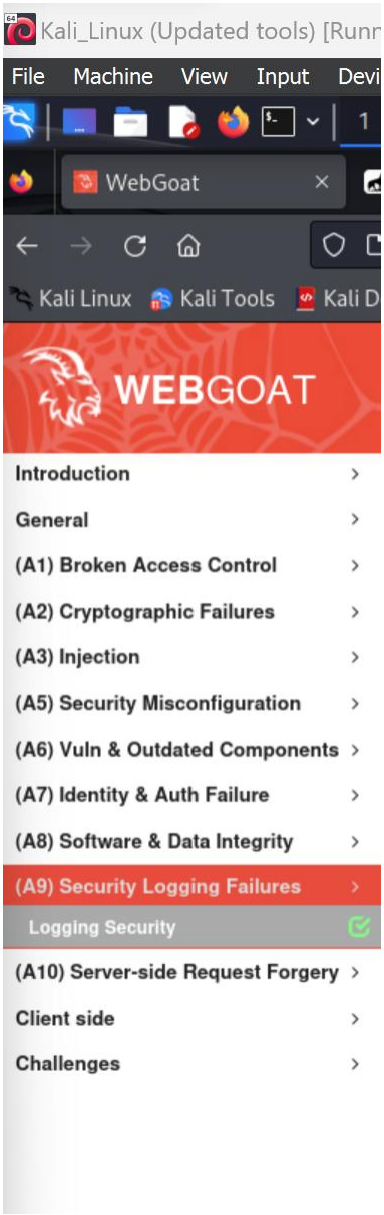


9.1 Insecure Deserialization

Summary:

Within this section we learned about what serialization is and how it can be used to do certain task that the developer did not intend to be used. Serialization is a process of turing object into a data format that can be restored later. Attackers do this to store or send data as part of a

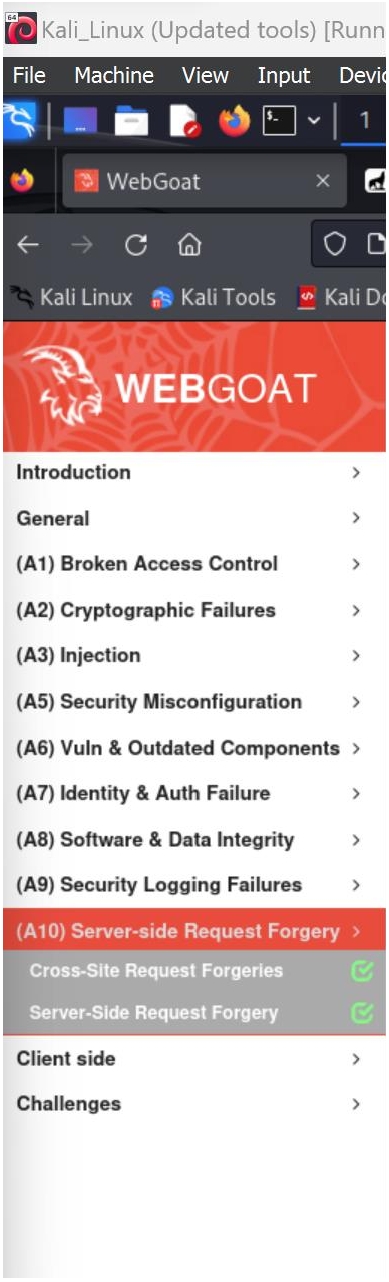
communication. On top of this you can also have a gadget chain even if they are rarer. A gadget chain is a dangerous action that itself can perform, however it is possible to have a gadget run multiple tasks on another gadget that run even more and so on and so forth until a dangerous action is ran.



10.1 Logging Security

Summary:

This lesson taught me that logging is very needed and very important for security because it can monitor apps and can also debug them, it can audit logs and even do security event monitoring. However, to make sure that your logs are secure you have to make sure that you keep sanitizing against spoofing attacks however there is more to log security. Another way to make sure you your data is safe from attacks do not log personal information or be very carful when you are doing it because people can use logs to spy on you.



11.1 Cross-Site Request Forgeries

Summary:

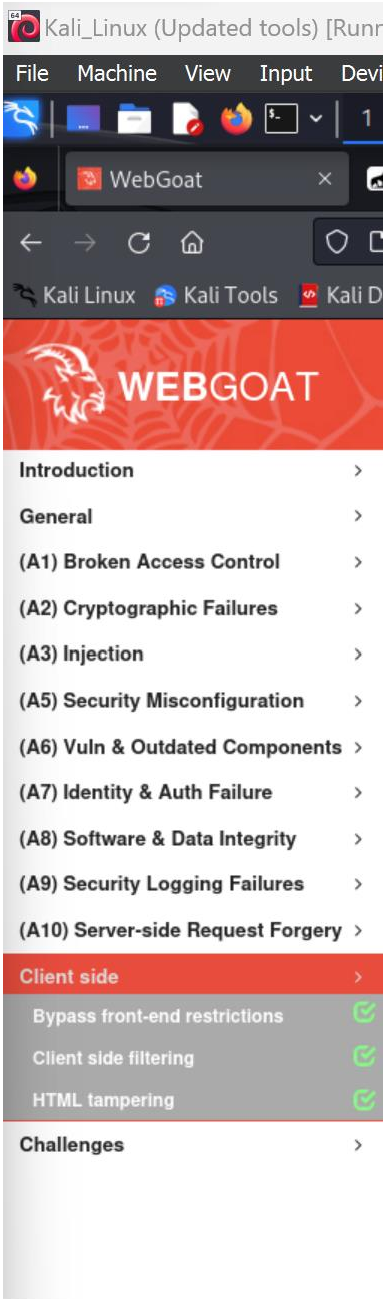
To understand this lesson, you have to learn that cross-site request forgery (CSRF) is a type of attack on a website where unauthorized commands are transmitted from a user's that the website trusts. This is a dangerous attack because unlike XSS attack a CSRF attack comes from a

user's that the website trusts and has less security towards that user. On of the main ways this attack is implemented is by sending an email to a user's and the users clicks on the email and then the email in the background will send a GET request to any open websites on the users' devices. To protect yourself form these attacks is make sure that you do not turn off the website protection for the most part they will protect you from a CSRF attack.

11.2 Server-Side Request Forgery

Summary:

In this module we went over SSRF (Server-Side Request Forgery) where the attacker can abuse the function of a server to make it read or update internal resources. By using the URL and carefully selecting it the attacker can read servers configurations like AWS metadata or HTTP enabled databases. To prevent these attacks, it recommended following the guidelines of using a whitelist pf allowed domains so that the webserver can grab the necessary resources. Also, any kind of input should be validated and rejected if it does not match the specification of the webserver, and if possible do not even accept user input from where webservers can receive resources.



12.1 Bypass front-end restrictions

Summary:

These next few lessons were short but still taught me important information. For example, in this lesson it taught me how users have a lot of control over the front-end of a web

application. For example, most browsers the user has complete or most control over the HTML part of the webpage to change values or restrictions to match their preferences. However, most webservers do have some sort of system to prevent users from sending any of their changes to the server like you cannot edit the script during runtime.

12.2 Client side filtering

Summary:

For this section it talked about Client side filtering and how it is always a good practice to send only information to the client they are supposed to have access to. In this lab we go over how to much information is being sent to the client and we have to exploit the information returned by the server in order to obtain information that we should not have access to.

12.3 HTML tampering

Summary:

Last Lesson in web goat is about how browsers have multiple options of how you want to display the content of a webserver to fit your preferences. This means that developers have to take into consideration that the values sent by the users has a good chance of being tampered with. The best way to make sure that this does not become a problem is to double check you original values with your database so that nothing has changed.