Cross-site scripting (XSS)

Previously, we explored a few types of malware.
Whether it's installed on an individual computer or a network server, all malicious software needs to be delivered to the target before it can work.
Phishing and other social engineering techniques are common ways for malware to be delivered.
Another way it's spread is using a broad class of threats known as web based exploits.

Web-based exploits are malicious code or behavior that's used to take advantage of coding flaws in a web application.
Cybercriminals target web-based exploits to obtain sensitive personal information.
Attacks occur because web applications interact with multiple users across multiple networks.
Malicious hackers commonly exploit this high level of interaction using injection attacks.

An injection attack is malicious code inserted into a vulnerable application.
The infected application often appears to work normally.
That's because the injected code runs in the background, unknown to the user.
Applications are vulnerable to injection attacks because they are programmed to receive data inputs.
This could be something the user types, clicks, or something one program is sharing with another.
When coded correctly, applications should be able to interpret and handle user inputs.

For example, let's say an application is expecting the user to enter a phone number.
This application should validate the input from the user to make sure the data is all numbers and not more than ten digits.
If the input from the user doesn't meet these requirements, the application should know how to handle it.
Web apps interact with multiple users across many platforms.
They also have a lot of interactive objects like images and buttons.
This makes it challenging for developers to think of all the ways they should sanitize their input.

A common and dangerous type of injection attack that's a threat to web apps is cross-site scripting.
Cross site scripting, or XSS, is an injection attack that inserts code into a vulnerable website or web application.
These attacks are often delivered by exploiting the two languages used by most websites, HTML and JavaScript.
Both can give cybercriminals access to everything that loads on the infected web page.
This can include session cookies, geolocation, and even webcams and microphones.

There are three main types of cross-site scripting attacks reflected, stored, and DOM-based.

A reflected XSS attack is an instance where a malicious script is sent to the server and activated during the server's response.

A common example of this is the search bar of a website.
In a reflected XSS attack, criminals send their target a web link that appears to go to a trusted site.
When they click the link, it sends a HTTP request to the vulnerable site server.
The attacker script is then returned or reflected back to the innocent user's browser.
Here, the browser loads the malicious script because it trusts the server's response.
With the script loaded, information like session cookies are sent back to the attacker.

In a stored XSS attack, the malicious script isn't hidden in a link that needs to be sent to the server.
Instead a stored XSS attack is an instance when malicious script is injected directly on the server.
Here, attackers target elements of a site that are served to the user.
This could be things like images and buttons that load when the site is visited.
Infected elements activate the malicious code when a user simply visits the site.
Stored XSS attacks can be damaging because the user has no way of knowing the site is infected beforehand.

Finally there's DOM-based XSS. DOM stands for Document Object Model, which is basically the source code of a website.
A DOM-based XSS attack is an instance when malicious script exists in the web page a browser loads. Unlike reflected XSS,
these attacks don't need to be sent to the server to activate.

In a DOM-based attack, a malicious script can be seen in the URL.
In this example, the website's URL contains parameter values.
The parameter values reflect input from the user.
Here, the site allows users to select color themes.
When the user makes a selection, it appears as part of the URL.
In a DOM-based attack, criminals change the parameter that suspecting an input.
For example, they could hide malicious JavaScript in the HTML tags.
The browser would process the HTML and execute the JavaScript.

Hackers use these methods of cross-site scripting to steal sensitive information.
Security analysts should be familiar with this group of injection attacks.
However, they're not the only ones, as we'll discover next time.

Revision #1
Created 27 August 2023 14:02:44 by naruzkurai
Updated 27 August 2023 14:04:16 by naruzkurai