Front end vulnerabilities

Thursday, January 18, 2024 1:01 PM

Sensitive Data Exposure

front end components dont pose a
direct threat to back end but
they do put end user in danger

attacks on admin users result
in unauth access, sens data,
service disrupt, etc.

most web pen testing is backend but
good to know frontend for finding
hidden accesses

Sensitive data exposure - availability of
sensitive data in clear-text to
end user

usually in source code

Usualy in source code

could find login creds, hashes or
sens data in comments or
in external JS code being
imported

also exposed links / directories

looking at source code should be one
of first things we do

important to classify data types and
what can/cant be seen on client
side

good to use JS obfuscation to reduce
chances of exposing code

HTML injection

Some user input never makes it to the backend and is entirely processed and rendered on front end

HTML injection - unfiltered user input is displayed on page

- retrieving previously submitted code like user comment from backend
- directly displaying input to front end

example of malicious HTML code could be fake login form could be fake login form

web defacing = insert malicious ads, change appearance, or change page

for front end examples, refreshing the page usually fixes any input

Cross site scripting (XSS)

inject JS code to be executed on client side

if we can execute code on victim's machine we could potentially gain access to account or machine

3 types of XSS

3 types of XSS

- reflected XSS - user input is displayed on the page after processing

  search result or error message

- Stored XSS - user input stored in back end and displayed on retrieval

  posts or comments

- DOM XSS - user input directly shown in browser and written to an HTML DOM object

  vulnerable username or page title

an example of DOM based XSS:

```
#"><img src=/ onerror=alert(document.
```

#"><img src=/ onerror=ale
cookie)>

browser processes input and it is considered as new DOM, JS is executed

Cross-site Request Forgery (CSRF)

CSRF may use XSS to perform queries and an API will call on web app that victim is already auth to

common attack to gain higher privilege access to web app is to use JS payload that changes user's pword

victim views payload on vulnerable

victim views payload on [illegible]
page (like a malicious comment)

Prevention

Sanitation - remove special characters and non-standard chars

Validation - ensure provided input matches expected format

also important to sanitize displayed output

WAF also help