**Esercitazione 16/12/2024**

Sempre sulla SQL injection.

Possiamo usare il form come un oracolo eseguendo una serie di tentativi.

Nel suo sito prova andiamo ad <http://.../blind> e usiamo il form dell’ip come oracolo, in che modo?

Prima cerchiamo un ID studente, quando ne troviamo uno che esiste testiamo cercando “1 and 1=1”.

Si chiama blindSQL injection perché la pagina web non ci restituisce una tabella ma un risultato in base ad un booleano, quindi, non vediamo quello che stiamo cercando.

Se volessi scoprire il grado dello studente, ad esempio, come facciamo a formulare questa domanda all’oracolo? Cerchiamo “1 and (SELECT grade from students where id = 1)= ‘A’” in pratica stiamo chiedendo se il grado di Alice è uguale ad A e lui userà quel booleano, se il booleano è vero allora la nostra domanda è giusta.

Se sapessi già qualcosa sulla tabella potrei usare:   
“1 and (SELECT 1 from PRAGMA\_TABLE\_INFO(‘students’) where name like ‘h%’)=1)”  
(h% significa che inizia con h e può finire con qualsiasi cosa)

Tale esempio serve per ex-filtrare il nome di colonne che non conosco a priori ad esempio.

Per scoprire quello che mi serve devo usare un approccio a bruteforce, quindi, bisogna magari creare un programmino che restituisca la lettera più probabile in base alle lettere già inserite.

Ci sono anche altri tipo di attacco in modalità oracolo, come time-based e error-based.

Challenge overthewire ([OverTheWire: Wargames](https://overthewire.org/wargames/)) per esercitarsi noi su quelli NATAS che sono gli attacchi web 🡪 Possiamo utilizzare delle dispense create da noi o cheetsheet

SEVERAMENTE VIETATO SQLmap o altri tool automatizzati.

Dopo aver avviato un livello apriamo burp suite e leggiamo il codice sorgente della pagina.

Usiamo l’include di php, come? Su burp suite>>repeater>>GET /index.php ?page=/..

E’ possible usare i filtri di php (li troviamo online come php filter) come ulterirore vulnerabilità, come ad esempio il base64 encoder.

Assumendo di essere nella home:  
GET /index.php ?page=php://filter/convert.base64.encode/resource=index.php così facendo troverò tutto il cod sorgente in base64

Lo copio in | base64 -d > index.php

Su port swigger in path traversal ci sono altre challenge da fare.

Per trovare la vulnerabilità basta usare l’intruder di burp con un dizionario di caratteri speciali (da trovare).

Trovare tutte le funzioni php, java, python e node.js che ci permettono di eseguire codice.

$\_REQUEST a differeza di $\_GET o $\_POST (roba di php) significa che si possono usare sia GET che POST.

Nel NATAS 9 l’attacco si basa su command injection (NB: su linux ; termina e # commenta).

Riprendere tutti i comandi di linux, usare sito acktriks per altre esercitazioni e port swigger