

# SQL Injection

Mitigações



# Mitigações



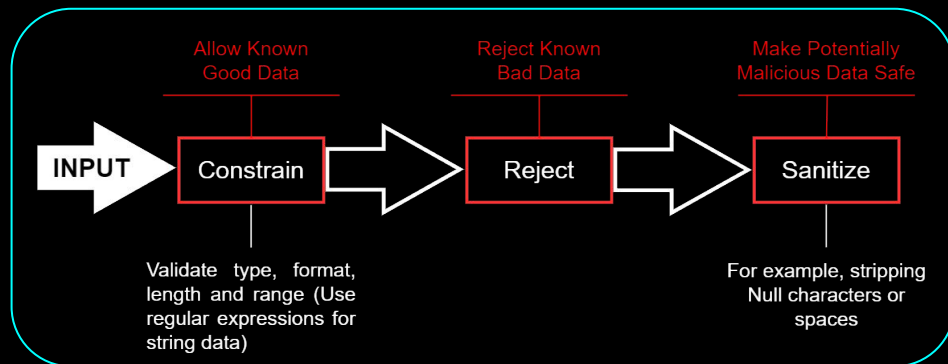
- Validação de Input do Usuário
- Tratamento de Erros
- Privilégios de Usuários dos Bancos de Dados
- Encriptação de Informação Sensível
- Implementar Controles de Segurança
- Monitorização Contínua



# Validação de Input do Usuário

Devem ser implementadas validações tanto na aplicação front-end como na aplicação back-end que permitam mitigar esta vulnerabilidade. Isto porque as validações aplicadas na aplicação front-end podem facilmente ser contornadas com um proxy como o Burp Proxy.

- **Validar que o input do usuário é válido antes de fazer a requisição.**  
Ex: validar formato de email, tipo de informação enviada, tamanho, etc..

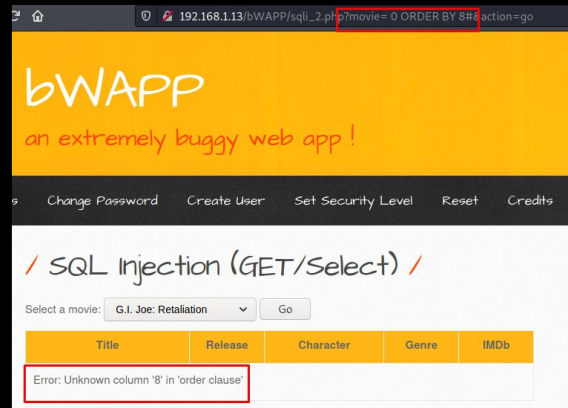
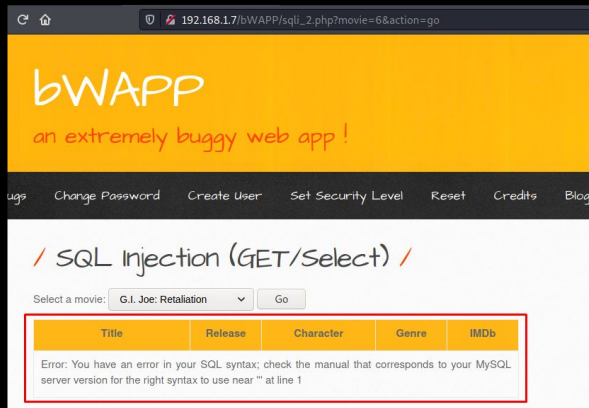


- **Validar que o input do usuário não tem conteúdo malicioso.** Ex: caracteres inválidos, conteúdo que possam representar código SQL, Javascript, etc..
- **Tratar o input do usuário para que o seu conteúdo malicioso não cause dano à aplicação.** Ex: Colocar o conteúdo ' OR 1=1# como String na query SQL, para que este conteúdo não seja interpretado como código SQL.

**Importante:** Isto pode ser feito através de [Prepared Statements](#).

# Tratamento de Erros

Tratar erros gerados na aplicação back-end de forma a que estes não sejam mostrados pela aplicação front-end para o usuário. Se os erros SQL não forem mostrados pela aplicação ao usuário, será muito mais difícil identificar uma vulnerabilidade de SQL Injection e explorar a mesma.



# Privilégios de Usuários do Banco de Dados

Quando são criadas contas de usuários no servidor do banco de dados, estas devem ser criadas com o mínimo de privilégios possíveis. Quando é criada uma conta de usuário no servidor do banco de dados, esta deve apenas ter as permissões necessárias para acessar e manipular o banco de dados da respectiva aplicação web. Assim sendo, se esta aplicação web for vulnerável a SQL Injection, o hacker consegue apenas acessar ao respectivo banco de dados e não a outros bancos de dados destinadas a outras aplicações web.

## Mutillidae

- **Usuário do Banco de dados:** mutillidae\_user / pass321
- **Privilégios:** Todos os privilégios, apenas para o banco de dados mutillidae

## bWAPP

- **Usuário do Banco de dados:** root / password123
- **Privilégios:** Todos os privilégios para todos os bancos de dados do servidor de bancos de dados.

- Se um hacker explorar SQL Injection na aplicação web bWAPP, consegue acessar aos registos de todos os bancos de dados (Mutillidae, DVWA, bWAPP, etc..).

- Se um hacker explorar SQL Injection na aplicação web Mutillidae, consegue apenas acessar aos registos do banco de dados da aplicação mutillidae.



# Encriptação de Informação Sensível

192.168.1.8/multilidae/index.php?page=info&username=wemake 'union select 1,username,password,is\_admin

Switch to SOAP Web Service version Switch to XPath version

[15/09/16] [WARNING] reflective value(s) found and filtering out  
Database: multilidae  
Table: accounts  
[24 entries]

cid	is_admin	lastname	username	firstname	password	mysignature
1	TRUE	Administrator	admin	System	adminpass	gtt r0n?
2	TRUE	Crenshaw	adrian	Adrian	s0mepassword	Zombie Films Rock!
3	FALSE	Pentest	John	John	monkey	I like the smell of confunk
4	FALSE	Drulin	Jeremy	Jeremy	password	d1373 1337 speak
5	FALSE	Galbraith	bryce	Bryce	password	I Love SAMS
6	FALSE	WTF	samurai	Samurai	samurai	Carving fools
7	FALSE	Rome	jim	Jim	password	Rome is burning
8	FALSE	Hill	bobby	Bobby	password	Hank is my dad
9	FALSE	Lion	simba	Simba	password	I am a super-cat
10	FALSE	Evil	drevel	Dr.	password	Preparation H
11	FALSE	Evil	scotty	Scotty	password	Scotty do
12	FALSE	Callipari	cal	John	password	C-A-T-S Cats Cats Cats
13	FALSE	Wall	John	John	password	Do the Duggie!
14	FALSE	Johnson	kevin	Kevin	42	Doug Adams rocks
15	FALSE	Kennedy	dave	Dave	set	Bet on S.E.T. FTW
16	FALSE	Pester	patches	Patches	tortoise	meow
17	FALSE	Paws	rocky	Rocky	stripes	treats?
18	FALSE	Tomes	tIm	Tim	lamaster53	Because reconnaissance is hard to spell
19	TRUE	Baker	Abaker	Aaron	SoSecret	Muffin tops only
20	FALSE	Pan	Pan	Peter	NotTelling	Where is Tinker?
21	FALSE	Hook	C0ok	Captain	JollyRoger	Gator-hater
22	FALSE	Jardine	James	James	iCiders	Occupation: Researcher
23	FALSE	Skoudis	ed	Ed	pentest	Commandline KungFu anyone?
24	NULL	NULL	wemake	NULL	wemake	wemake

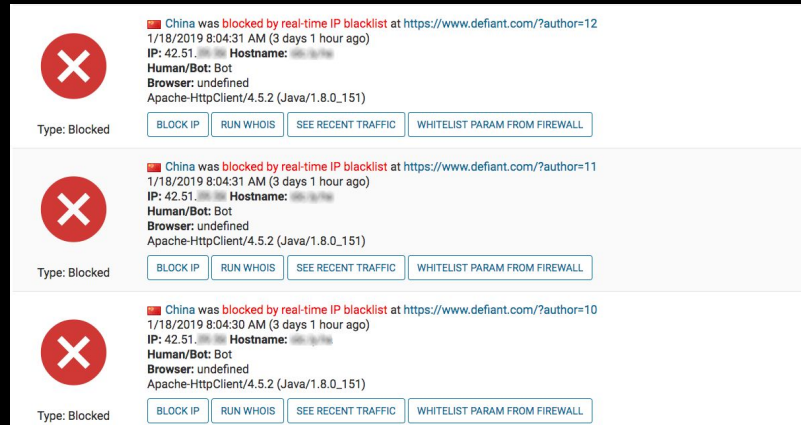
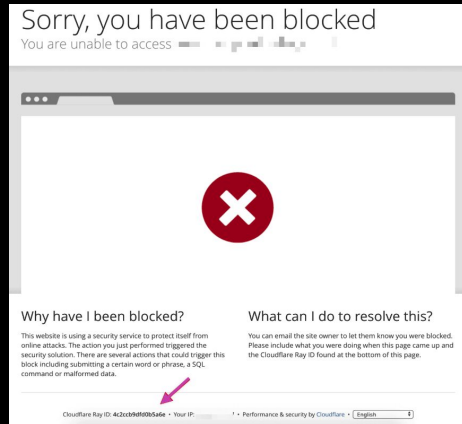
Um dos principais objetivos de explorar SQL Injection é obter credenciais de contas de usuários. Embora não seja frequente, ainda existem casos de informação sensível como passwords de usuários não encriptadas guardadas no banco de dados.

Nestes casos, recomenda-se a implementação de um mecanismo de encriptação de passwords no banco de dados.

Assim, se houverem acessos não autorizados ao banco de dados, o indivíduo que faz o acesso não consegue visualizar a password de outros usuários.

# Implementar Controles de Segurança

A implementação de controlos de segurança como Web Application Firewalls (WAF) ou Intrusion Prevention Systems (IPS) pode também fazer parte da mitigação desta vulnerabilidade. Este tipo de controlos ajuda não só a prevenir a exploração de vulnerabilidades de SQL Injection, como muitas outras vulnerabilidades que possam existir na aplicação web.



# Monitorização Contínua

A monitorização constante dos eventos é um fator fundamental para uma análise de risco eficaz nos sistemas de qualquer empresa. Sistemas de monitorização avançados permitem a recolha e análise destes eventos em tempo real, de forma a refletir a exposição a ameaças dos sistemas das empresas.



## SecurityWall

Software de monitorização e análise de eventos desenvolvido pela empresa WeSecure.

### **Capacidade de Monitorização engloba:**

- Monitorização de recursos;
- Integridade de ficheiros;
- Disponibilidade dos ativos;

### **São detectadas ameaças tais como:**

- Ataques ao sistema e ataques web;
- Tentativas de intrusão;
- Erros de aplicações;
- Anomalias do sistema;
- Violações de políticas de segurança, etc..