MLS PostgreSQL

Joe Conway
joe.conway@crunchydata.com
mail@joeconway.com

Crunchy Data

January 21, 2016

MLS PostgreSQL

- What is Multi-Level Security (MLS)?
- Security Level
  - Sensitivity
  - Category

Technologies

- PostgreSQL
  - Row Level Security (RLS)
  - Customized sepgsql
- Red Hat Enterprise Linux
  - Networking
  - SELinux
  - Custom SELinux Policy

Caveats

- Talk covers development system, not production
- Permissive mode
- Work In Process

Acknowledgements

- Stephen Frost
- Adam Brightwell
- Mike Palmiotto
- Jason O'Donnell
- Red Hat and Lockheed Martin
- Others . . .

Agenda

- Solution Components
  - RLS
  - SELinux
  - sepgsql
- Implementation
  - Installation and Configuration
    - Operating System
    - Networking
    - SELinux
    - PostgreSQL
  - Database schema/DDL
- Results

Row Level Security

- New feature in PostgreSQL 9.5
- Enabled on per-table basis
- Enforced with POLICY
  - USING expression (old row)
  - WITH CHECK expression (new row)

Row Level Security - Typical Example

```
CREATE USER bob;
CREATE USER alice;

CREATE TABLE t1 (id int primary key, f1 text, app_user text);
INSERT INTO t1 VALUES(1,'a','bob');
INSERT INTO t1 VALUES(2,'b','alice');
ALTER TABLE t1 ENABLE ROW LEVEL SECURITY;
CREATE POLICY P ON t1 USING (app_user = current_user);
GRANT SELECT ON t1 TO public;
```

Row Level Security - Typical Example

```
SELECT * FROM t1;
 id | f1 | app_user
----+----+----------
  1 | a  | bob
  2 | b  | alice

SET SESSION AUTHORIZATION bob;
SELECT * FROM t1;
 id | f1 | app_user
----+----+----------
  1 | a  | bob

SET SESSION AUTHORIZATION alice;
SELECT * FROM t1;
 id | f1 | app_user
----+----+----------
  2 | b  | alice
```

Security Enhanced Linux

- SELinux: Mandatory Access Control (MAC)
- Versus: Discretionary Access Control (DAC)
- Enforced in kernel space
- Managed via Reference Policy
  - Targeted Policy
  - MLS Policy
- Customized via Policy Modules

https://people.redhat.com/duffy/selinux/selinux-coloring-book_A4-Stapled.pdf

MLS Reference Policy

- Based on Bell-LaPadula model
  - Read-down
  - Write-up
- Modified for Write-equals

Security Context

- <user>:<role>:<domain>:<sensitivity>:<category>
  - <user> = SElinux user
  - <role> = SElinux role
  - <domain> = type
  - <sensitivity> = low to high, e.g. s0, s1, ...s15
  - <category> = compartmentalization label
- <level> = <sensitivity>:<category>
- Examples

```
dbs6_u:dbclient_r:dbclient_t:s0
system_u:object_r:sepgsql_table_t:s0-s15:c0.c1023
```

Security Level

- s0-s15
  - Represents a range of sensitivities
  - Can be defined with aliases
  - Hierarchical dominance is defined
- c0.c1023
  - Represents a group of categories
  - Can be defined with aliases
  - No hierarchical dominance

Security Access Decision

- Subject Context (PostgreSQL user)
- Object/Target Context (table, row, etc.)
- Permission (e.g. select, update, etc.)
- Type Enforcement
  - Subject type needs requested permission on object type, e.g.:
  - `allow postgresql_t sepgsql_table_type : db_table { create drop ... select update insert delete lock };`
- Sensitivity
  - Subject must dominate Object
  - e.g. s5:c1.c5 dominates s3:c42
- Category
  - Subject must include Object category
  - e.g. s5:c1.c5 does not include s3:c42

sepgsql Extension

- PostgreSQL supports `SECURITY LABEL` command
- Label Provider uses the label
- Security label used for SELinux Object context
- Customized with additional functionality
  - User mapping database user to SELinux user
  - Subject context transition based on postgres user and netlabel
  - `sepgsql_check_row_label()`
  - `sepgsql_create_row_label()`

Object label support

- Standard
  - SCHEMA
  - TABLE, VIEW, COLUMN
  - SEQUENCE
  - FUNCTION
- Custom
  - ROW

sepgsql_check_row_label(arg1 [, arg2])

- Object context: arg1 - row security_label
- Subject context: client - SELinux user+netlabel
- Permission Type: default `select`, otherwise arg2:
  - `select`, `insert`, `update`, `delete`
  - `relabelfrom`, `relabelto`
- Access decision: SELinux

sepgsql_check_row_label(arg1 [, arg2])

```
select sepgsql_getcon();
           sepgsql_getcon
-------------------------------------
 dbs5_u:dbclient_r:dbclient_t:s5:c1

SELECT
 sepgsql_check_row_label
 ('system_u:object_r:sepgsql_table_t:s0') as s0sel,
 sepgsql_check_row_label
 ('system_u:object_r:sepgsql_table_t:s6') as s6sel;
 s0sel | s6sel
-------+-------
 t     | f
```

sepgsql_check_row_label(arg1 [, arg2])

```
select sepgsql_getcon();
           sepgsql_getcon
-------------------------------------
 dbs5_u:dbclient_r:dbclient_t:s5:c1

SELECT
 sepgsql_check_row_label
 ('system_u:object_r:sepgsql_table_t:s0','delete') as s0del,
 sepgsql_check_row_label
 ('system_u:object_r:sepgsql_table_t:s5','delete') as s5del,
 sepgsql_check_row_label
 ('system_u:object_r:sepgsql_table_t:s5:c1','delete') as s5c1del;
 s0del | s5del | s5c1del
-------+-------+---------
 f     | f     | t
```

sepgsql_create_row_label(table_oid)

- Object context: Table security label
- Subject context: client - SELinux user+netlabel
- Derives security_label context, typically used for a row

```
CREATE OR REPLACE FUNCTION get_table_label(tableoid oid)
RETURNS text AS $$
 SELECT label FROM pg_seclabels WHERE objoid = tableoid
 AND objtype = 'table'
$$ LANGUAGE sql;

\x
SELECT get_table_label('t1'::regclass) AS tcontext,
 sepgsql_getcon() AS scontext,
 sepgsql_create_row_label('t1'::regclass) AS security_label;
-[ RECORD 1 ]--+-----------------------------------------------
tcontext       | system_u:object_r:sepgsql_table_t:s0-s15:c0.c1023
scontext       | dbs5_u:dbclient_r:dbclient_t:s5:c1
security_label | dbs5_u:object_r:sepgsql_table_t:s5:c1
```

sepgsql_create_row_label(table_oid)

```
\x
SELECT get_table_label('t1'::regclass) AS tcontext,
 sepgsql_getcon() AS scontext,
 sepgsql_create_row_label('t1'::regclass) AS security_label;
-[ RECORD 1 ]--+-----------------------------------------------
tcontext       | system_u:object_r:sepgsql_table_t:s0-s15:c0.c1023
scontext       | dbs6_u:dbclient_r:dbclient_t:s6:c1
security_label | dbs6_u:object_r:sepgsql_table_t:s6:c1
```

Operating System

- Download and install Red Hat or CentOS 7.2
- Talk based on Gnome desktop configuration
- Install additional packages

Operating System - Packages

```
yum install epel-release
yum update

# install PGDG 9.5 rpms
# http://www.postgresql.org/download/linux/redhat/#yum
yum install http://yum.postgresql.org/9.5/redhat/\
rhel-7-x86_64/pgdg-redhat95-9.5-2.noarch.rpm
yum install postgresql95\*

# install selinux rpms
yum install netlabel_tools selinux-policy-mls \
    libsemanage-python policycoreutils-python \
    setools-libs setools-console xinetd selinux-policy-devel
```

Networking

- Interfaces
  - admin subnet
  - subnet per security level
- Routes
- netlabel
- sshd
- firewalld

Networking - Interfaces

```
cat /etc/sysconfig/network-scripts/ifcfg-enp3s0
TYPE="Ethernet"
BOOTPROTO="none"
DEVICE="enp3s0"
ONBOOT="yes"
IPADDR="192.168.4.20"
PREFIX="24"
IPADDR1="192.168.5.20"
PREFIX1="24"
IPADDR2="192.168.6.20"
PREFIX2="24"
IPADDR3="192.168.7.20"
PREFIX3="24"
IPADDR4="192.168.8.20"
PREFIX4="24"
GATEWAY="192.168.4.1"
DNS1="192.168.4.1"
[...]
```

Networking - Routes

```
route
Kernel IP routing table
Destination  Gateway      Genmask        Flg Met Ref Use Iface
default      192.168.4.1  0.0.0.0        UG  100   0   0 enp3s0
192.168.4.0  0.0.0.0      255.255.255.0  U   100   0   0 enp3s0
192.168.5.0  0.0.0.0      255.255.255.0  U   100   0   0 enp3s0
192.168.6.0  0.0.0.0      255.255.255.0  U   100   0   0 enp3s0
192.168.7.0  0.0.0.0      255.255.255.0  U   100   0   0 enp3s0
192.168.8.0  0.0.0.0      255.255.255.0  U   100   0   0 enp3s0
```

Networking - netlabel

- Allows security context labeling of packets
- Based on incoming network
- Shown configurations specific to environment
  $\Rightarrow$ modify as appropriate for target environment

```
cat >> /etc/netlabel.rules << \EOF
# Custom rules
map del default
map add default address:0.0.0.0/0 protocol:unlbl
cipsov4 add pass doi:5 tags:5

# Add local ethernet interfaces and loopback
map add default address:192.168.4.20 protocol:cipsov4,5
map add default address:192.168.5.20 protocol:cipsov4,5
map add default address:192.168.6.20 protocol:cipsov4,5
map add default address:192.168.7.20 protocol:cipsov4,5
map add default address:192.168.8.20 protocol:cipsov4,5
map add default address:127.0.0.0/8 protocol:cipsov4,5
EOF
```

Networking - netlabel

```
cat >> /etc/netlabel.rules << \EOF

# Accept unlabeled traffic by default.
unlbl accept on

# Add incoming IP address ranges
# Include entry for each virtual interface

# admin subnet
unlbl add interface:enp3s0 address:192.168.4.0/24 \
  label:system_u:object_r:netlabel_peer_t:s0-s15:c0.c1023

# lowest level interface (e.g. unclassified)
unlbl add interface:enp3s0 address:192.168.5.0/24 \
  label:system_u:object_r:netlabel_peer_t:s0
EOF
```

Networking - netlabel

```
cat >> /etc/netlabel.rules << \EOF

# next level interface (e.g. classified)
unlbl add interface:enp3s0 address:192.168.6.0/24 \
  label:system_u:object_r:netlabel_peer_t:s4:c1

# next level interface (e.g. secret)
unlbl add interface:enp3s0 address:192.168.7.0/24 \
  label:system_u:object_r:netlabel_peer_t:s5:c1

# top level interface (e.g. top secret)
unlbl add interface:enp3s0 address:192.168.8.0/24 \
  label:system_u:object_r:netlabel_peer_t:s6:c1

# catch all
unlbl add interface:enp3s0 address:0.0.0.0/0 \
  label:system_u:object_r:netlabel_peer_t:s0
EOF
```

Networking - netlabel

```
# Enable Netlabel.
systemctl enable netlabel.service

# Start Netlabel.
systemctl start netlabel.service

# note, if you ever have to modify
# /etc/netlabel.rules then do
systemctl stop netlabel.service
netlabel-config reset
systemctl start netlabel.service
```

Networking - sshd

- Switch from normal sshd service to sshd socket service
- This allows netlabel to work for ssh connections

```
vi /lib/systemd/system/sshd.socket
   # Add to [Socket] section
   # comment this out if netlabel is not working
   # or else connections will be refused
   SELinuxContextFromNet=true

# swap enabled service
systemctl disable sshd.service
systemctl enable sshd.socket

# swap active service
systemctl stop sshd.service
systemctl start sshd.socket
```

Networking - firewalld

- Add firewall rule to allow PostgreSQL connections

```
# add postgres rule
firewall-cmd --permanent --add-service=postgresql

# activate it
firewall-cmd --reload
```

SELinux - conf File

- Red Hat 7 SELinux defaults
  - `targeted` reference policy
  - `enforcing` mode
- Switch to
  - `mls` reference policy
  - `permissive` mode

```
vi /etc/selinux/config
   SELINUX=permissive
   SELINUXTYPE=mls

# ensure selinux is currently permissive
setenforce 0
```

SELinux - Relabeling

- Configure run-level
- Map login OS user
- Arrange to relabel at next boot
  - autorelabel only modifies type portion of existing contexts
  - -F option will force relabel entire context

```
# Set default run level to multi-user.target or graphical.target
systemctl set-default multi-user.target

# map normal user to staff_u
semanage login -a -s staff_u jconway

echo "-F" > /.autorelabel
reboot

sestatus
```

SELinux - Custom Modules

- Install custom policy modules

```
cd /opt/src/mls/crunchy-mls-selinux-policy
make
make install
```

SELinux - Custom Modules

- Verify expected roles exist

```
seinfo -adomain -r|grep -E "^   db"
   dbguest_r
   dbstaff_r
   dbadm_r
   dbown_r
   dbsec_r
   dbsu_r
   dbclient_r
```

SELinux - Create Users

- Create selinux users
- Will map later to database users

```
semanage user -a -R "dbadm_r dbstaff_r dbsu_r" \
                           -r "s0-s15:c0.c1023" postgres_u
semanage user -a -R "dbown_r"    -r "s0-s15:c0.c1023" dbown_u
semanage user -a -R "dbstaff_r"  -r "s0-s15:c0.c1023" dbstaff_u
semanage user -a -R "dbguest_r"  -r "s0"              dbguest_u
semanage user -a -R "dbclient_r" -r "s0"              dbs0_u
semanage user -a -R "dbclient_r" -r "s0-s4:c0.c9"     dbs4_u
semanage user -a -R "dbclient_r" -r "s0-s5:c0.c200"   dbs5_u
semanage user -a -R "dbclient_r" -r "s0-s6:c0.c1023"  dbs6_u
```

SELinux - User Default Contexts

- Configure default contexts
- Map context transition

```
cat > /etc/selinux/mls/contexts/users/postgres_u << \EOF
object_r:netlabel_peer_t:s0     dbadm_r:dbadm_t:s0
sysadm_r:sysadm_t:s0            dbadm_r:dbadm_t:s0
staff_r:staff_t:s0              dbadm_r:dbadm_t:s0
EOF
cat > /etc/selinux/mls/contexts/users/dbs0_u << \EOF
object_r:netlabel_peer_t:s0     dbclient_r:dbclient_t:s0
sysadm_r:sysadm_t:s0            dbclient_r:dbclient_t:s0
staff_r:staff_t:s0              dbclient_r:dbclient_t:s0
EOF
[...]
cat > /etc/selinux/mls/contexts/users/dbs6_u << \EOF
object_r:netlabel_peer_t:s0     dbclient_r:dbclient_t:s0
sysadm_r:sysadm_t:s0            dbclient_r:dbclient_t:s0
staff_r:staff_t:s0              dbclient_r:dbclient_t:s0
EOF
```

PostgreSQL - Initialize and Start

- Initialize PostgreSQL
- Verify you can log in

```
# initdb to create new cluster
postgresql95-setup initdb

# enable the service
systemctl enable postgresql-9.5

# start the service
systemctl start postgresql-9.5

# check status
systemctl status postgresql-9.5
sudo -u postgres psql -l
```

PostgreSQL - Host Based Authentication

- Configure access

```
su - postgres
psql -c "alter user postgres password 'postgres'"

# comment out existing lines in pg_hba.conf
sed -i -r 's/^(local|host)/#\1/g' $PGDATA/pg_hba.conf

# edit pg_hba.conf: allow local and approved subnets
cat >> $PGDATA/pg_hba.conf << \EOF
local all all                 md5
host  all all 127.0.0.1/32    md5
host  all all ::1/128         md5
host  all all 192.168.0.0/16  md5
EOF

# make them take effect
exit
systemctl reload postgresql-9.5
```

PostgreSQL - Database Users

- Connect and create some postgres roles

```
psql -U postgres << \EOF
create user dbguest password 'dbguest';
create user dbclient password 'dbclient';
create role dbown nologin;
EOF
```

PostgreSQL - Custom Module

- Build and Configure custom sepgsql
- Adjust some normal PostgreSQL configuration too

```
cd /opt/src/mls/crunchy-selinux-pgsql
USE_PGXS=1 make
USE_PGXS=1 make install

cat >> /var/lib/pgsql/9.5/data/postgresql.conf << \EOF
listen_addresses = '*'
row_security = on
shared_preload_libraries = 'crunchy-selinux-pgsql'

sepgsql.enable_user_transition = on
sepgsql.default_selinux_user = 'dbguest_u'
sepgsql.force_rls = on
EOF
```

PostgreSQL - Custom Module

- Install custom sepgsql

```
systemctl stop postgresql-9.5

# Install custom sepgsql functions
su - postgres
for DBNAME in template0 template1 postgres
do
  postgres --single -F -c \
           exit_on_error=true $DBNAME \
  < /usr/pgsql-9.5/share/contrib/crunchy-selinux-pgsql.sql \
  > /dev/null
done
exit

systemctl start postgresql-9.5
systemctl status postgresql-9.5
```

PostgreSQL - Custom Module

- One more bit of custom configuration
- sepgsql-users.conf maps Postgres role to SELinux user
- Should be unnecessary as of PostgreSQL 9.6

```
cat > /var/lib/pgsql/9.5/data/sepgsql-users.conf << \EOF
postgres postgres_u
dbguest  dbguest_u
user1    dbs0_u
user2    dbs4_u
user3    dbs5_u
user4    dbs6_u
EOF
```

Create and Load Database

```
psql -h 192.168.4.20 -p 5432 -U postgres postgres \
     -c "create database mls"

# Next few slides show the important details herein
psql -h 192.168.4.20 -p 5432 -U postgres mls \
     -c "\i crunchy-mls-demo-setup.sql"
```

Create Demo Users

```
-- Create demo users
CREATE USER user1 WITH ENCRYPTED PASSWORD 'user1';
CREATE USER user2 WITH ENCRYPTED PASSWORD 'user2';
CREATE USER user3 WITH ENCRYPTED PASSWORD 'user3';
CREATE USER user4 WITH ENCRYPTED PASSWORD 'user4';
```

Table Definition

```
CREATE TABLE t1 (
  a int,
  b text,
  security_label text DEFAULT
  sepgsql_create_row_label('t1'::regclass::oid)
);

-- Grant permissions to table
GRANT ALL ON TABLE t1 TO user1, user2, user3, user4;

-- Enable Row Level Security on table.
ALTER TABLE t1 ENABLE ROW LEVEL SECURITY;
```

Table Definition

```
-- Create Row Level MLS policies.
CREATE POLICY mls_select ON t1 FOR SELECT
  USING (sepgsql_check_row_label(security_label));

CREATE POLICY mls_insert ON t1 FOR INSERT WITH CHECK
 (sepgsql_create_row_label('t1'::regclass::oid) = security_label);

CREATE POLICY mls_update ON t1 FOR UPDATE
 USING (sepgsql_check_row_label(security_label))
 WITH CHECK (sepgsql_check_row_label(security_label,'update'));

CREATE POLICY mls_delete ON t1 FOR DELETE
 USING (sepgsql_check_row_label(security_label,'delete'));
```

Sample Data

```
-- Seed table with sample data
INSERT INTO t1 VALUES
  (1, 'a', 'system_u:object_r:sepgsql_table_t:s0'),
  (2, 'b', 'system_u:object_r:sepgsql_table_t:s4:c1'),
  (3, 'c', 'system_u:object_r:sepgsql_table_t:s5:c1'),
  (4, 'd', 'system_u:object_r:sepgsql_table_t:s6:c1');
```

User Level Versus Subnet Level

```
# s0 user, s4 subnet
psql -h 192.168.6.20 -p 5432 -U user1 mls
Password for user user1:
psql: FATAL:  SELinux: unable to get default context for user: user1

# s0 user, s0 subnet
psql -qAt -h 192.168.5.20 -p 5432 -U user1 mls \
 -c "select sepgsql_getcon()"
Password for user user1:
dbs0_u:dbclient_r:dbclient_t:s0

# s6 user, s0 subnet
psql -qAt -h 192.168.5.20 -p 5432 -U user4 mls \
 -c "select sepgsql_getcon()"
Password for user user4:
dbs6_u:dbclient_r:dbclient_t:s0
```

SELECT on s0 Subnet

```
# s0 user, s0 subnet
psql -h 192.168.5.20 -p 5432 -U user1 mls \
 -c "select * from t1"
Password for user user1:
 a | b |            security_label
---+---+--------------------------------------
 1 | a | system_u:object_r:sepgsql_table_t:s0
(1 row)

# s6 user, s0 subnet
psql -h 192.168.5.20 -p 5432 -U user4 mls \
 -c "select * from t1"
Password for user user4:
 a | b |            security_label
---+---+--------------------------------------
 1 | a | system_u:object_r:sepgsql_table_t:s0
(1 row)
```

user4 SELECT on s6 Subnet

```
# s6 user, s6 subnet
psql -h 192.168.8.20 -p 5432 -U user4 mls \
 -c "select * from t1"
Password for user user4:
 a | b |             security_label
---+---+----------------------------------------
 1 | a | system_u:object_r:sepgsql_table_t:s0
 2 | b | system_u:object_r:sepgsql_table_t:s4:c1
 3 | c | system_u:object_r:sepgsql_table_t:s5:c1
 4 | d | system_u:object_r:sepgsql_table_t:s6:c1
(4 rows)
```

INSERT on s0 Subnet

```
# s0 user, s0 subnet
psql -h 192.168.5.20 -p 5432 -U user1 mls \
 -c "insert into t1(a,b) values (11,'a1') returning *"
Password for user user1:
 a  | b  |           security_label
----+----+------------------------------------
 11 | a1 | dbs0_u:object_r:sepgsql_table_t:s0
(1 row)

# s6 user, s0 subnet
psql -h 192.168.5.20 -p 5432 -U user4 mls \
 -c "insert into t1(a,b) values (41,'a1') returning *"
Password for user user4:
 a  | b  |           security_label
----+----+------------------------------------
 41 | a1 | dbs6_u:object_r:sepgsql_table_t:s0
(1 row)
```

INSERT on s6 Subnet

```
# s6 user, s6 subnet
psql -h 192.168.8.20 -p 5432 -U user4 mls \
 -c "insert into t1(a,b) values (441,'d1') returning *"
Password for user user4:
  a  | b  |            security_label
-----+----+--------------------------------------
 441 | d1 | dbs6_u:object_r:sepgsql_table_t:s6:c1
(1 row)
```

UPDATE on s0 Subnet

```
# s0 user, s0 subnet, s0 row
psql -h 192.168.5.20 -p 5432 -U user1 mls \
 -c "update t1 set b = 'a1a' where a = 11 returning *"
Password for user user1:
 a  |  b  |           security_label
----+-----+------------------------------------
 11 | a1a | dbs0_u:object_r:sepgsql_table_t:s0
(1 row)

# s6 user, s0 subnet, s0 row
psql -h 192.168.5.20 -p 5432 -U user4 mls \
 -c "update t1 set b = 'd1d' where a = 41 returning *"
Password for user user4:
 a  |  b  |           security_label
----+-----+------------------------------------
 41 | d1d | dbs6_u:object_r:sepgsql_table_t:s0
(1 row)
```

UPDATE on s6 Subnet

```
# s6 user, s6 subnet, s6 row
psql -h 192.168.8.20 -p 5432 -U user4 mls \
 -c "update t1 set b = 'd1d' where a = 441 returning *"
Password for user user4:
  a  |  b  |            security_label
-----+-----+--------------------------------------
 441 | d1d | dbs6_u:object_r:sepgsql_table_t:s6:c1
(1 row)

# however...s6 user, s6 subnet, s0 row
psql -h 192.168.8.20 -p 5432 -U user4 mls \
 -c "update t1 set b = 'd1d1' where a = 41 returning *"
Password for user user4:
ERROR:  new row violates row-level security policy for table "t1"
```

Change Row Security Level

```
# s6 user, s0 subnet, change row to s6
psql -h 192.168.5.20 -p 5432 -U user4 mls \
 -c "update t1 set security_label =
     'dbs6_u:object_r:sepgsql_table_t:s6:c1'
     where a = 41 returning *"
Password for user user4:
ERROR:  new row violates row-level security policy for table "t1"

# s6 user, s6 subnet, change row to s6
psql -h 192.168.8.20 -p 5432 -U user4 mls \
 -c "update t1 set security_label =
     'dbs6_u:object_r:sepgsql_table_t:s6:c1'
     where a = 41 returning *"
Password for user user4:
 a  |  b  |            security_label
----+-----+--------------------------------------
 41 | d1d | dbs6_u:object_r:sepgsql_table_t:s6:c1
(1 row)
```

DELETE on s6 Subnet

```
# s6 user, s6 subnet, delete rows at s6
psql -h 192.168.8.20 -p 5432 -U user4 mls \
 -c "delete from t1 where a > 9 returning *"
Password for user user4:
  a   |  b   |            security_label
------+------+--------------------------------------
 441 | d1d | dbs6_u:object_r:sepgsql_table_t:s6:c1
  41 | d1d | dbs6_u:object_r:sepgsql_table_t:s6:c1
(2 rows)

DELETE 2
```

DELETE on s6 Subnet - Results

```
# s6 user, s6 subnet, show rows at s6 and below
psql -h 192.168.8.20 -p 5432 -U user4 mls \
 -c "select * from t1"
Password for user user4:
 a  |  b  |              security_label
----+-----+-------------------------------------
  1 | a   | system_u:object_r:sepgsql_table_t:s0
  2 | b   | system_u:object_r:sepgsql_table_t:s4:c1
  3 | c   | system_u:object_r:sepgsql_table_t:s5:c1
  4 | d   | system_u:object_r:sepgsql_table_t:s6:c1
 11 | a1a | dbs0_u:object_r:sepgsql_table_t:s0
(5 rows)
```

DELETE on s0 Subnet

```
# s6 user, s0 subnet, delete rows at s0
psql -h 192.168.5.20 -p 5432 -U user4 mls \
 -c "delete from t1 where a > 9 returning *"
Password for user user4:
 a  |  b  |           security_label
----+-----+------------------------------------
 11 | a1a | dbs0_u:object_r:sepgsql_table_t:s0
(1 row)

DELETE 1
```

Questions?

Thank You!
mail@joeconway.com