Label based Mandatory Access Control on PostgreSQL

NEC Europe Ltd,
SAP Global Competence Center
KaiGai Kohei <kohei.kaigai@eu.nec.com>
SE-PostgreSQL got merged in v9.1

NEW! contrib/sepgsql
History of development

- **Sep-2006** Launch development of SE-PostgreSQL based on v8.2.x
- **Apr-2007** First post to psql-hackers, after 2 weeks of feature freeze
- **Mar-2007** SELinux Symposium 2007
- **Nov-2007** METI Japan gave an award due to SE-PostgreSQL
- **May-2008** PGcon2008 – SE-PostgreSQL
- **Jul-2008** Development Cycle for v8.4
  - Too large to review
- **Jul-2009** Development Cycle for v9.0
  - Steps to consensus up to the current design
- **May-2010** PostgreSQL Developer Summit
- **Sep-2010** SECURITY LABEL statement got merged
- **Jan-2011** contrib/sepgsql got merged
- **May-2011** PGcon2011 – Label based MAC on PostgreSQL
- **Jun-2011** 1st Commit Fest of v9.2 development cycle
Today’s Agenda

- Overview of label based MAC
- New features in v9.1
- Our challenges to v9.2
Characteristics of MAC

DAC (discretionary access control) : Owner decide access control rules

MAC (mandatory access control) : A centralized security policy decides access control rules
**Data Flow Control**

- **Keep confidential data in confidential domain**
  - No Read-Up
  - No Write-Down (Only same level)
  - Restriction to malicious internals

- **Background**
  - TCSEC (Orange book; 1983)
  - ISO/IEC15408 (CC: Common Criteria)
A module that suggests its access control decision

Three characteristics
- **Always invoked**
- Tamperproof
- Small enough

SELinux performs as reference monitor in Linux kernel
Analogy in Linux and PostgreSQL

<table>
<thead>
<tr>
<th>SELinux</th>
<th>SE-PostgreSQL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object manager</td>
<td>Filesystem</td>
</tr>
<tr>
<td>Objects being referenced</td>
<td>file, directory, device file, ...</td>
</tr>
<tr>
<td>Way to request accesses</td>
<td>System call</td>
</tr>
<tr>
<td>DAC</td>
<td>Filesystem permission</td>
</tr>
<tr>
<td>MAC</td>
<td>LSM &amp; SELinux</td>
</tr>
</tbody>
</table>
Decision making of SELinux (1/2)

Access control logic like a function

- Subject
- Object
- Action

Input: read, write, select, ...

Access Control Logic

Output: Allowed, Denied

Decision:
- Filesystem Permission
- Database ACL
- SELinux
- others...?

- • Owner ID
- • Permission Bits
- • ACL
- • Security Label

- • User/Group ID
- • Superuser Priv
- • Security Label
Decision making of SELinux (2/2)

- The way to identify Subject/Object
  - Path name?
  - Owner ID?
  - Security Label

- Security Label as a universal way for identification
  Example)
  ```
  system_u:system_r:postgresql_t:s0
  system_u:object_r:sepgsql_ro_table_t:s0
  ```

- Example of security policy
  ```
  allow staff_t sepgsql_ro_table_t : db_table { select };
  ```

- SELinux uses white-list criteria.
- SELinux community provides general set of rules in default.
### OT: source code of the default security policy

<table>
<thead>
<tr>
<th>Part of the “policy/modules/services/postgresql.te” at the default security policy</th>
</tr>
</thead>
</table>

```c
policy_module(postgresql, 1.12.1)
    :
  type sepgsql_schema_t;
postgresql_schema_object(sepgsql_schema_t)
    :
  type sepgsql_table_t;
postgresql_table_object(sepgsql_table_t)
    :
  allow sepgsql_admin_type sepgsql_schema_type:
      db_schema { create drop getattr setattr relabelfrom relabelto search add_name remove_name };
  allow sepgsql_client_type sepgsql_schema_t:db_schema {getattr search };

  allow sepgsql_admin_type sepgsql_table_type:
      db_table { create drop getattr setattr relabelfrom relabelto lock };
  allow sepgsql_admin_type sepgsql_table_t:db_table {getattr select update insert delete lock };
  allow sepgsql_admin_type sepgsql_table_t:db_column {getattr use select update insert };
```

```c
```
System-wide consistency in Access control

Human user

Login

bash

vi

psql

classified information

domain of classified processes

Filesystem

X-window

inter process communication channels

 SELinux

Security Policy

PostgreSQL

X-window

Networks

psql

vi

bash

unclassified information

domain of unclassified processes

Human user

Login
Today’s Agenda

- Overview of label based MAC
- New features in v9.1
- Our challenges to v9.2
Features needed to support Label based MAC

- **Security Label**
  - mechanism to associate a short text with a particular database object
  - something like xattr in filesystem cases

- **Security Hook**
  - mechanism to acquire control on strategic points of the code
  - something like LSM in Linux kernel cases

- **Intermediation with SELinux**
  - mechanism to deliver a pair of security labels into SELinux in kernel, and prevents violated accesses according to its decision
v9.1 New Features (1/3) – SECURITY LABEL

**Overview**
- It enables to assign a text identifier of database objects.
- It allows security modules to reference security label of a particular object.

**Limitations**
- Shared database objects are not supported, right now.
- Tuples in user-defined tables are not supported, right now.

SECURITY LABEL ON TABLE my_example FOR 'selinux'
IS 'system_u:object_r:sepgsql_ro_table_t:s0';
OT: Labeled Networking

Labeled Networking
- SELinux provides `getpeercon(3)` to get security label of the peer process.
- Kernel & IKE daemon were enhanced to exchange security labels.
  - supported on kernel-2.6.18 or later, ipsec-tools 0.72 or later

Use case in SE-PostgreSQL
- It obtains security label of the peer process on the authentication hook.
- Peer security label is applied to subject’s label on access control decision.
Overview
- It enables 3rd party modules to acquire control on strategic points of the code.
  - E.g) Just after creation of the object for default labeling.
- The object_access_hook informs event type and object identifiers.

Limitations
- Only OAT_POST_CREATE event type is supported, right now.
  - ✓ May need OAT_CREATE, OAT_ALTER, ...
- Only object identifiers are informed via this hook, right now.
**Overview**

- It performs as intermediation between PostgreSQL and SELinux
  - PostgreSQL ... user Id, object Id,
  - SELinux ... security label, object class and permission

**Limitations**

- only DML permissions are checked, right now
- default security labels on schemas, tables, columns and procedures
Today’s Agenda

- Overview of label based MAC
- New features in v9.1
- Our challenges to v9.2
Limitation in v9.1, and Challenges to v9.2

- Frequent system-call invocations
  - Add access control decision cache

- No security label on shared objects
  - Add pg_shseclabel catalog, and extend SECURITY LABEL

- No DDL Permission checks
  - Extend object_access_hook to take arguments
  - Put object_access_hook around existing DDL checks

- Row-level access control
  - Fix leaky VIEWs problem
  - Extend security label on user-defined tables
v9.2 challenges (1/3) – Userspace access vector cache

Overview
- uavc keeps access control decision recently used; that allows to reduce number of system call invocations.

Challenges
- Cache invalidation on security policy reloaded on kernel-side
  - Linux 2.6.38 already support selinux status page.
v9.2 challenges (2/3) – DDL Permissions

Overview
- It allows to check permissions on DDL commands also.

Challenges
- Larger number of strategic points than DML support
- object_access_hook with additional arguments

postgres=# ALTER TABLE drink OWNER TO ymj;
LOG: SELinux: denied { setattr } ¥
scontext=unconfined_u:unconfined_r:unconfined_t:s0 ¥
tcontext=system_u:object_r:sepgsql_table_t:s0:c0 ¥
tclass=db_table name=drink
ERROR: SELinux: security policy violation
v9.2 challenges (3/3) – Row-level security

Overview
- Filter out rows based on security policy and labels of individual tuples

Challenges
- Fix the problem known as Leaky-VIEWs
- Security label support for user-defined tables
- Query rewriter to append security-policy function
- Interaction with system catalog

```sql
postgres=# SELECT security_label, * FROM drink;

| security_label                                      | id | name | price |
|-----------------------------------------------------+----+-------+-------|
| system_u:object_r:sepgsql_table_t:s0                |  1 | coke  |   150 |
| system_u:object_r:sepgsql_table_t:s0                |  2 | fanta |   130 |
| system_u:object_r:sepgsql_table_t:s0:c0             |  3 | beer  |   200 |
| system_u:object_r:sepgsql_table_t:s0:c1             |  4 | sake  |   240 |
| system_u:object_r:sepgsql_table_t:s0:c2             |  5 | juice |   180 |

(5 rows)
```
Future Vision (1/2) – Role based access control

- SQL with confined privileges
- Only DDL, DML not allowed
- Execute
- Log File
- pg_dump/pg_restore
- System Log Administrator
- Database Administrator

internet
customer
Future Vision (2/2) – Secure multi-tenancy

SELinux performs as a logical-wall between security domains.

http://www.blue.com/
http://www.red.com/
http://www.green.com/
Summary

- **Overview of MAC**
  - Data flow control and Reference monitor concept
  - SE-PostgreSQL enables to deploy RDBMS within DFC scheme.

- **Features in v9.1**
  - SECURITY LABEL
  - Object access hooks
  - contrib/sepgsql

- **Challenges to v9.2**
  - Userspace access vector cache
  - DDL Permissions
  - Row-level access control
Any Questions?
Thank you!
Empowered by Innovation

NEC