LAPP/SELinux

A secure web application stack using SE-PostgreSQL

KaiGai Kohei <kaigai@ak.jp.nec.com>

NEC OSS Promotion Center

Self Introduction

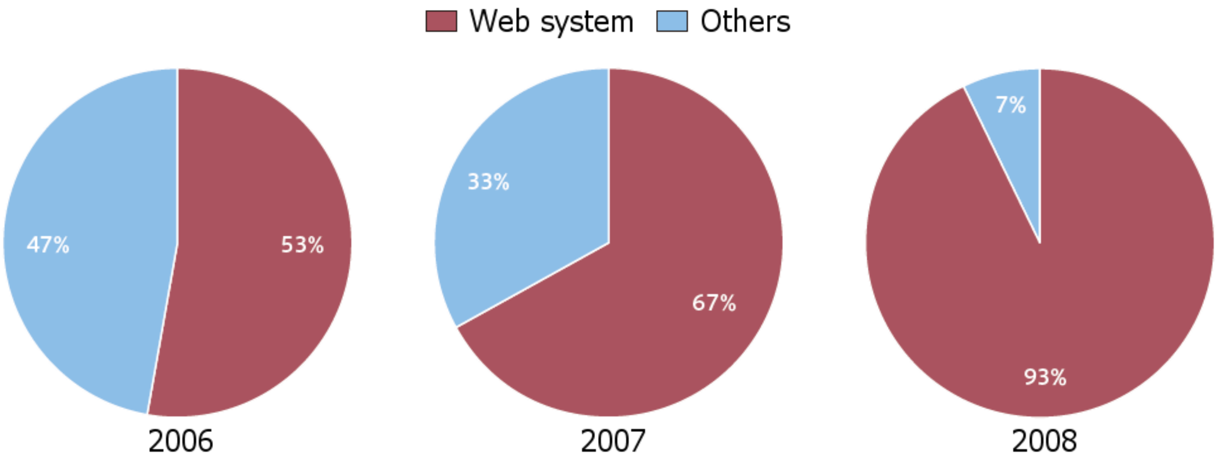
SELECT * FROM pg_developers WHERE name = 'KaiGai'

- Job NEC OSS Promotion Center, for 7 years
- Contributions
  - SMP Scalability Improvement of SELinux
  - Lead project to port SELinux into embedded platform
  - Development of SE-PostgreSQL
  - Access control support of large object, and so on...
- Interest Web system's security

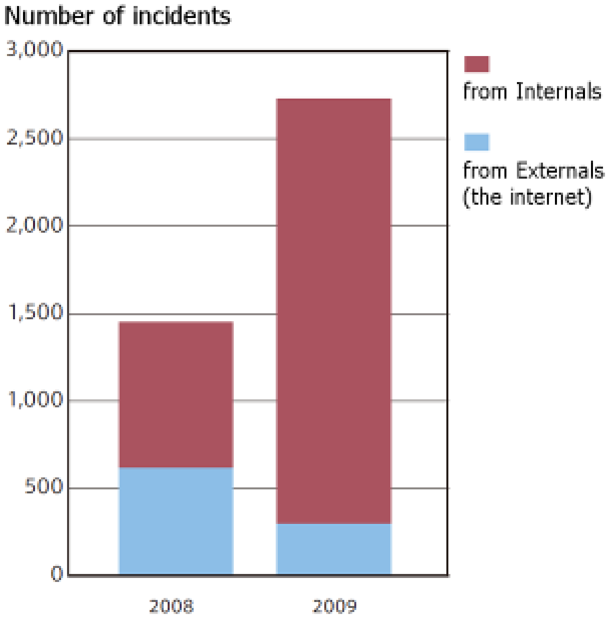
Agenda

1. Backgrond
2. SE-PostgreSQL
3. Apache/SELinux plus
4. Demonstration
5. Future Plans

Security nightmare in web systems



Targets of significant security incidents



(Reference: JSOC analysis report of the incursion trend, vol.12, vol.14, LAC)

- Rapid increasing of attacks to web systems
- More threats from Internals, rather than Externals

➡ What technology can improve the situation?

LAPP - A typical web application stack

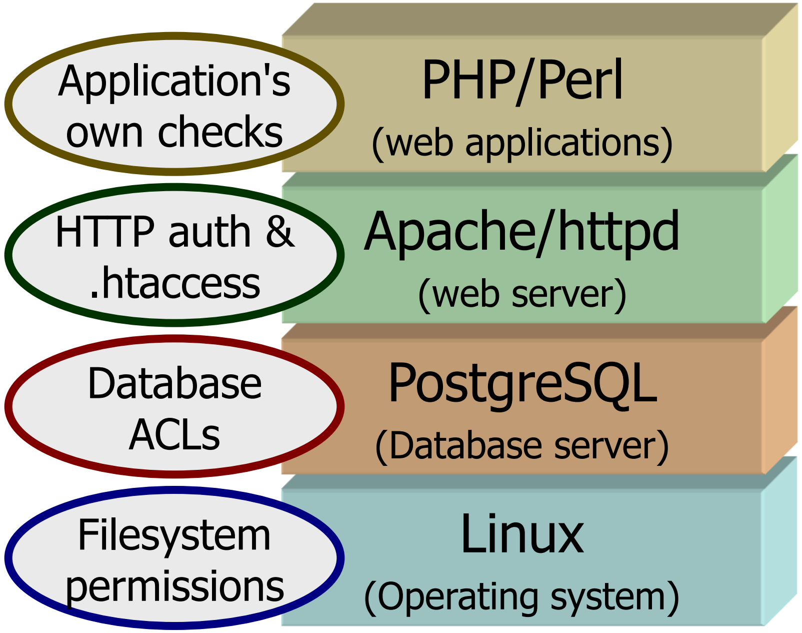
LAPP

- Linux, Apache, PostgreSQL, PHP/Perl

Concerns in security

- Each layer has its own security mechanism
- Web-users are not mapped to users in OS/DB

An information asset in DB being invisible might be visible in Filesystem



Application's own checks

PHP/Perl
(web applications)

HTTP auth & .htaccess

Apache/httpd
(web server)

Database ACLs

PostgreSQL
(Database server)

Filesystem permissions

Linux
(Operating system)

OS/DB layer could not distingiush actual users, so all the security burdens are pushed to web-app's

Software stack like a symphony,
But ....

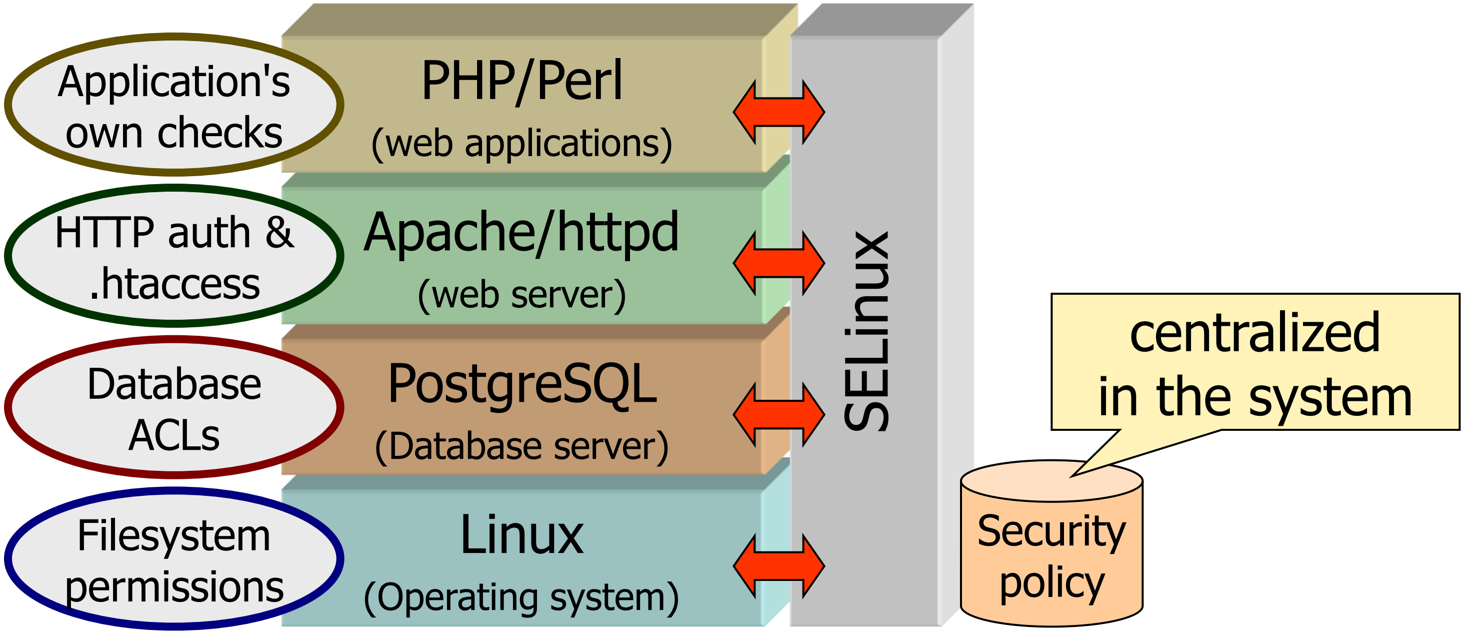
Lack of conductor

LAPP/SELinux - concept

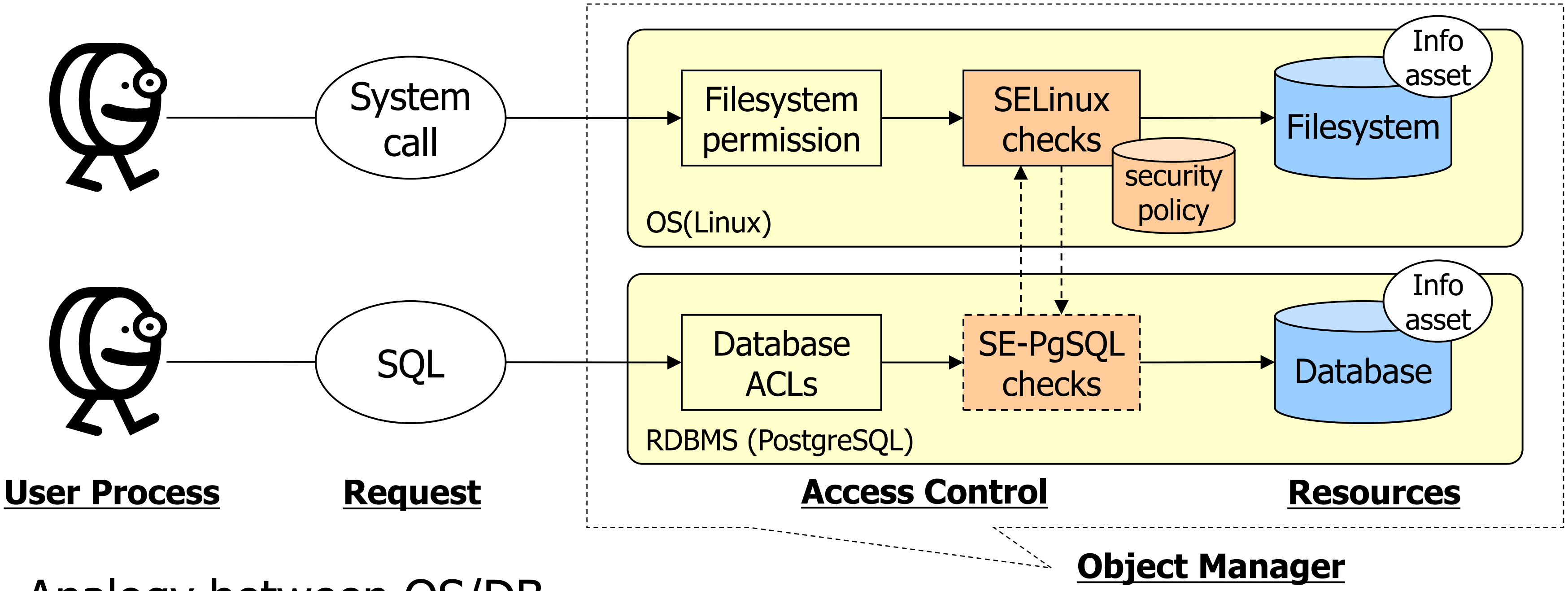
SELinux performs as conductor

- System-wide privileges are assigned to all the users
- DB controls accesses based on the centralized policy

➡ It ensures **least-privilege** and **consistency** in access control.



Perspective from the model (1/2)



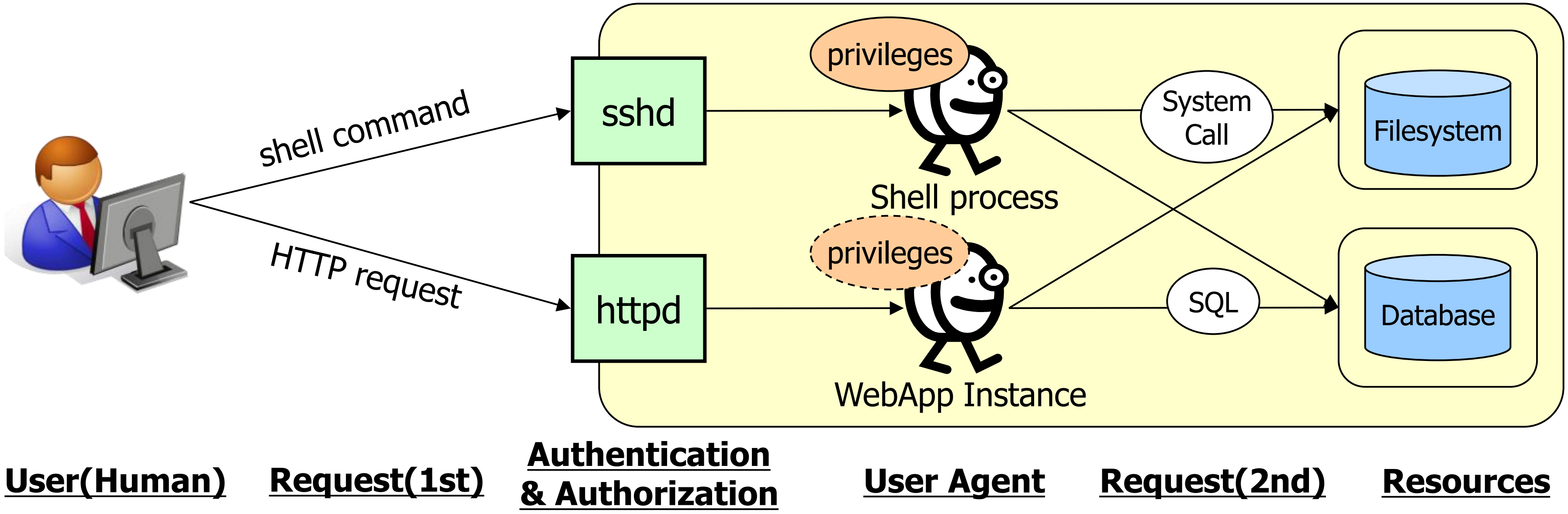
Analogy between OS/DB

- Differences in the way to store and access information assets
- System-call for Filesystem, SQL for Database

Role of access control

- It decides whats are allowed or disallowed between users and resources, and controls the given requests based on the decision.
- ➡ Same basis (security policy) ensures system-wide consistency.

Perspective from the model (2/2)



Analogy between shell and web

- User is a human; An user-agent performs instead of himself.
- User-agent must have correct privileges reflecting the actual human.

Role of authentication & anthorization

- It identifies the human connected, and assigns their privileges.
  - sshd assignes user/group-id on the login shell before the execution.
  - Apache does not change privileges of the web-application instance.

LAPP/SELinux - components

SE-PostgreSQL

- A built-in enhancement of PostgreSQL
- Additional permission checks on the given queries according to the decision of SELinux
- It ensures consistency in access controls

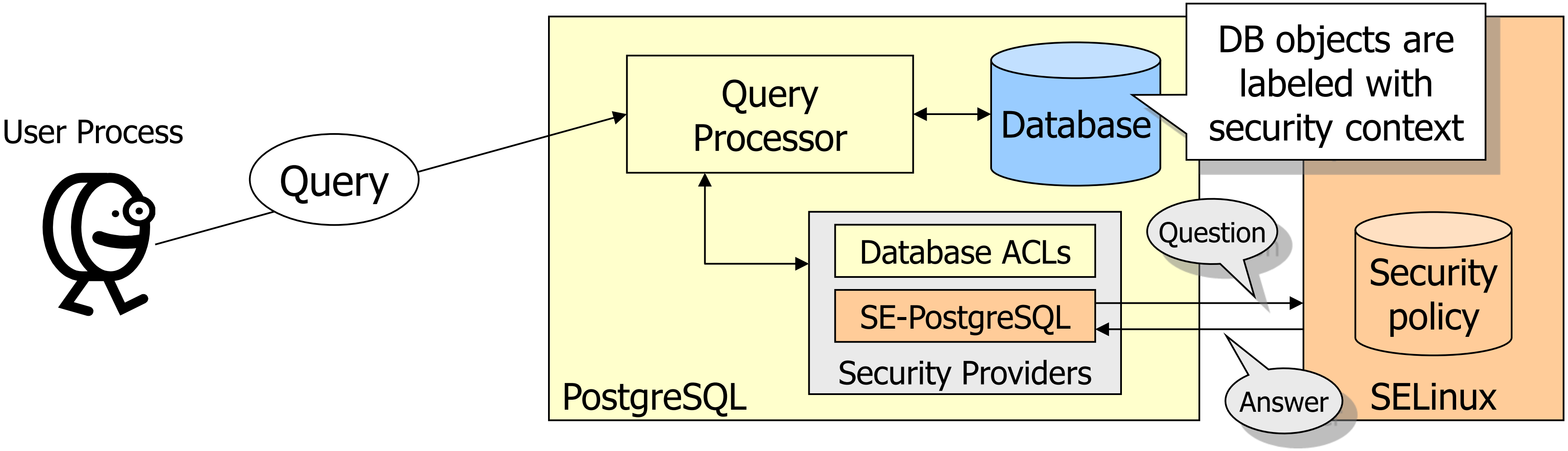
Apache/SELinux Plus

- A loadable module of the Apache/httpd 2.2.x
- It assignes a security context of the contents handler based on http authentication.
- It ensures least-privilege in access control; with utilization of OS/DB

Agenda

1. Backgrond
2. SE-PostgreSQL
3. Apache/SELinux plus
4. Demonstration
5. Future Plans

Architecture of SE-PostgreSQL



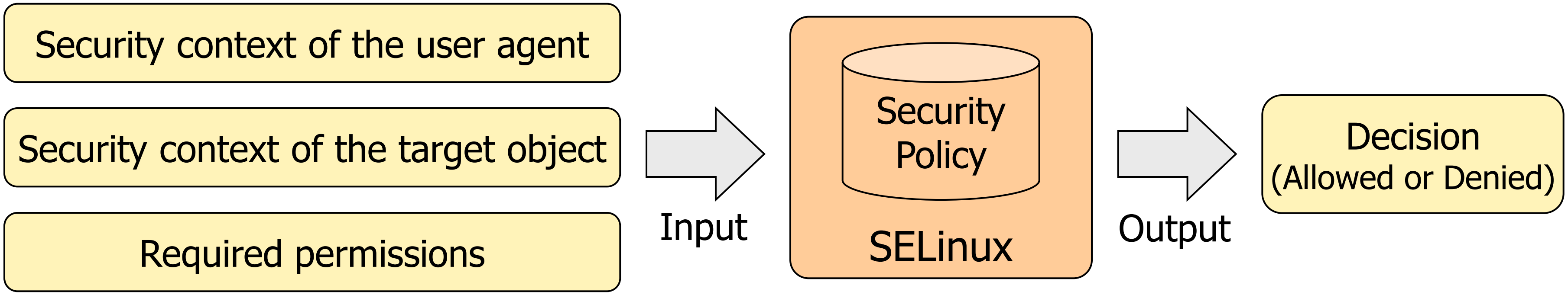
Security Providers

- Common entrypoints of access control features; like database ACLs.
- SE-PostgreSQL shall be an optional security provider.

SE-PostgreSQL

- It tells SELinux whether the given query is allowed to run;
  (Need to deliver a pair of security context of the client and objects)
- SELinux returns its decision,
  then SE-PostgreSQL raises an error if access violation.

Decision making in access controls


Security context of the user agent
Security context of the target object
Required permissions
Input
Security Policy
SELinux
Output
Decision (Allowed or Denied)

SELinux performs like a function

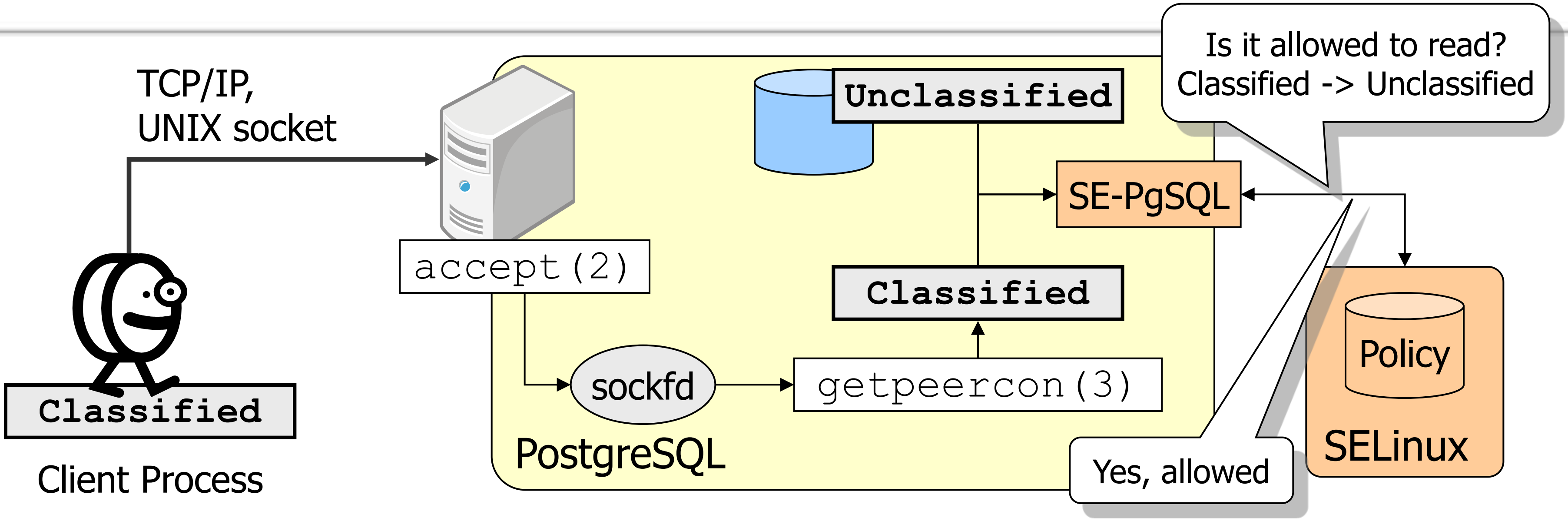
- It returns its decision for the given arguments.
- Kernel internally gives them to SELinux, and follows its decision.
- Userspace application can also utilize the mechanism, as long as it can provide pair of the security context.

Security context

- A SELinux specified identifier of processes and any other objects.

Example)
```
system_u:system_r:httpd_t:s0
system_u:object_r:postgresql_db_t:s0
```

Security context of the client process



Labeled networks

- SELinux provides an API to obtain security context of the peer process.

```
int getpeercon(int sockfd, security_context_t *con);
```

- IPsec daemon exchanges the security context of peers prior to open the connection.
- Static fallback security context for non-SELinux'ed clients.

➡ It allows to identify the client process using security context.

Security context of the database objects (1/2)

```
postgres=> SELECT security_label, * FROM drink;
             security_label              | id |   name   | price
-----------------------------------------+----+----------+--------
 system_u:object_r:sepgsql_table_t:s0    |  1 | water    |    110
 system_u:object_r:sepgsql_table_t:s0    |  2 | tea      |    130
 system_u:object_r:sepgsql_table_t:s0:c0 |  3 | coke     |    130
 system_u:object_r:sepgsql_table_t:s0:c1 |  4 | coffee   |    180
(4 rows)
```

"security_label" system column

- It represents the security context of tuples.
- The tuple of **`pg_class`** shows properties of table,
  so it means the security context of the table, for example.

Default security context

- On insertion, the default one shall be assigned based on the policy.
- User can also provide an explicit one, instead of the default.

Security context of the database objects (2/2)

ALTER xxx SECURITY LABEL TO

- It allows to change security context of database objects.
- Use **UPDATE** statements for tuples within user tables.

```
postgres=> ALTER TABLE t SECURITY LABEL TO
                 'user_u:object_r:sepgsql_ro_table_t:s0';
ALTER TABLE
```

ALTER TABLE xxx SET WITH/WITHOUT SECURITY LABEL

- It allows to strip 'security_label' system column, if not necessary.
- Reduce row-level control and storage consumption on the table.

```
postgres=> ALTER TABLE t SET WITHOUT SECURITY LABEL;
ALTER TABLE
postgres=> SELECT security_label, * FROM t;
ERROR:  column "security_label" does not exist
```

Usage of SE-PostgreSQL (1/2)

```
postgres=# CREATE TABLE customer
           (id integer primary key, name text, credit text);
postgres=# ALTER TABLE customer ALTER credit SECURITY LABEL TO
               'system_u:object_r:sepgsql_secret_table_t:s0';
postgres=# INSERT INTO customer
               VALUES (1, 'kaigai', '1111-2222-3333-4444');
```

```
postgres=# SELECT * FROM customer;
LOG:  SELinux: denied { select } ¥
        scontext=staff_u:staff_r:staff_t:s0 ¥
        tcontext=system_u:object_r:sepgsql_secret_table_t:s0 ¥
        tclass=db_column name=customer.credit
ERROR:  SELinux: security policy violation
postgres=# SELECT id, name FROM customer;
 id |  name
----+--------
  1 | kaigai
(1 row)
```

Client was not allowed to select from the column labeled as **`sepgsql_secret_table_t`**

Usage of SE-PostgreSQL (2/2)

```
postgres=# SELECT security_label, * FROM;
                  security_label                  | id |   name   | price
--------------------------------------------------+----+----------+-------
 system_u:object_r:sepgsql_table_t:Unclassified     |  1 | water    |   100
 system_u:object_r:sepgsql_table_t:Classified       |  2 | coke     |   120
 system_u:object_r:sepgsql_ro_table_t:Classified    |  3 | juice    |   140
 system_u:object_r:sepgsql_ro_table_t:Unclassified  |  4 | coffee   |   180
 staff_u:object_r:sepgsql_table_t:Unclassified      |  5 | beer     |   240
```

On SELECT

- All the tuples are visible for Classified user,
  but Classified tuples are not visible Unclassified user.

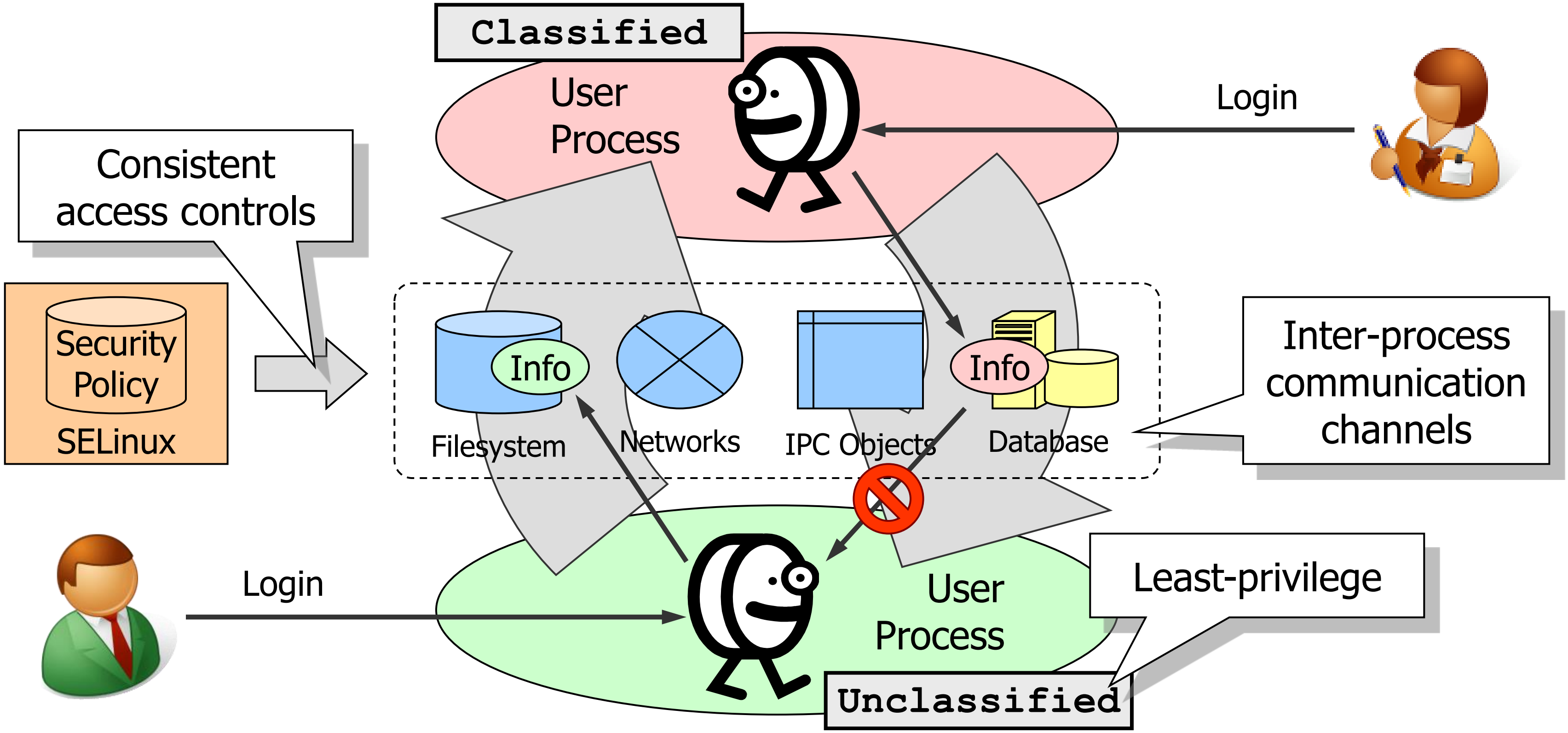
On UPDATE/DELETE

- Also, Classified tuples are updatable/deletable by Classified users.
- And, Read-only tuples are not updatable by confined users.

On INSERT

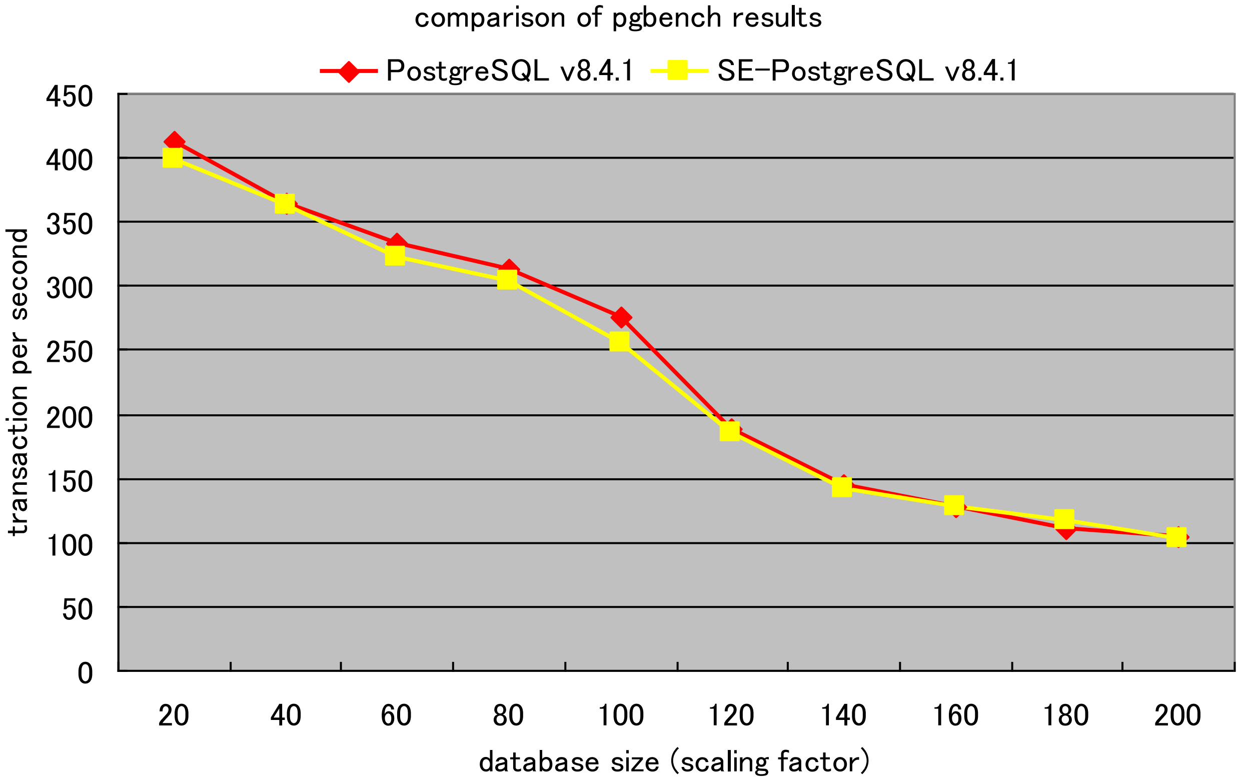
- A default security context shall be assigned on the new tuple,
  and checks privilege to insert it.

System-wide consistency in access controls



- SELinux provides its access control decision for ANY subsystems
  - ✓ Linux kernel enforces the decision on accesses to filesystem objects, and etc...
  - ✓ SE-PostgreSQL enforces the decision on accesses to database objects.
- Eventually, the centralized security policy controls all the accesses

Performance - SE-PostgreSQL

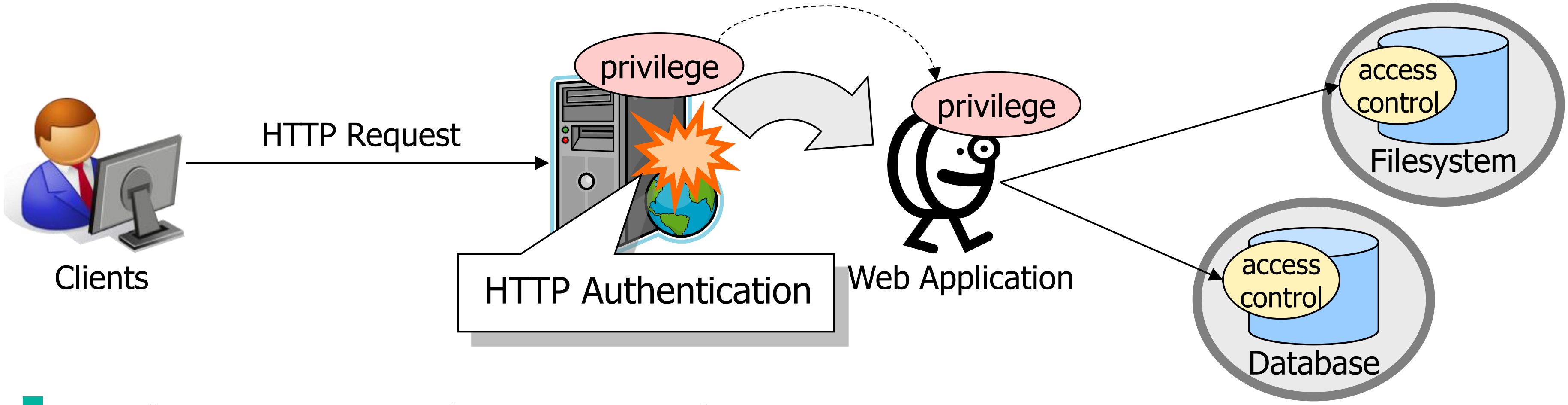


- 2~4% of trade-off in performance
  - userspace AVC minimizes the number of kernel invocations
- Environments
  - CPU Xeon (2.33GHz) Dual, Mem: 2GB (shared_buffer=512m)
  - measured by `pgbench -c 2 -t 200000`

Agenda

1. Backgrond
2. SE-PostgreSQL
3. Apache/SELinux plus
4. Demonstration
5. Future Plans

Who's privileges should be checked?



Authentication, but no authorization

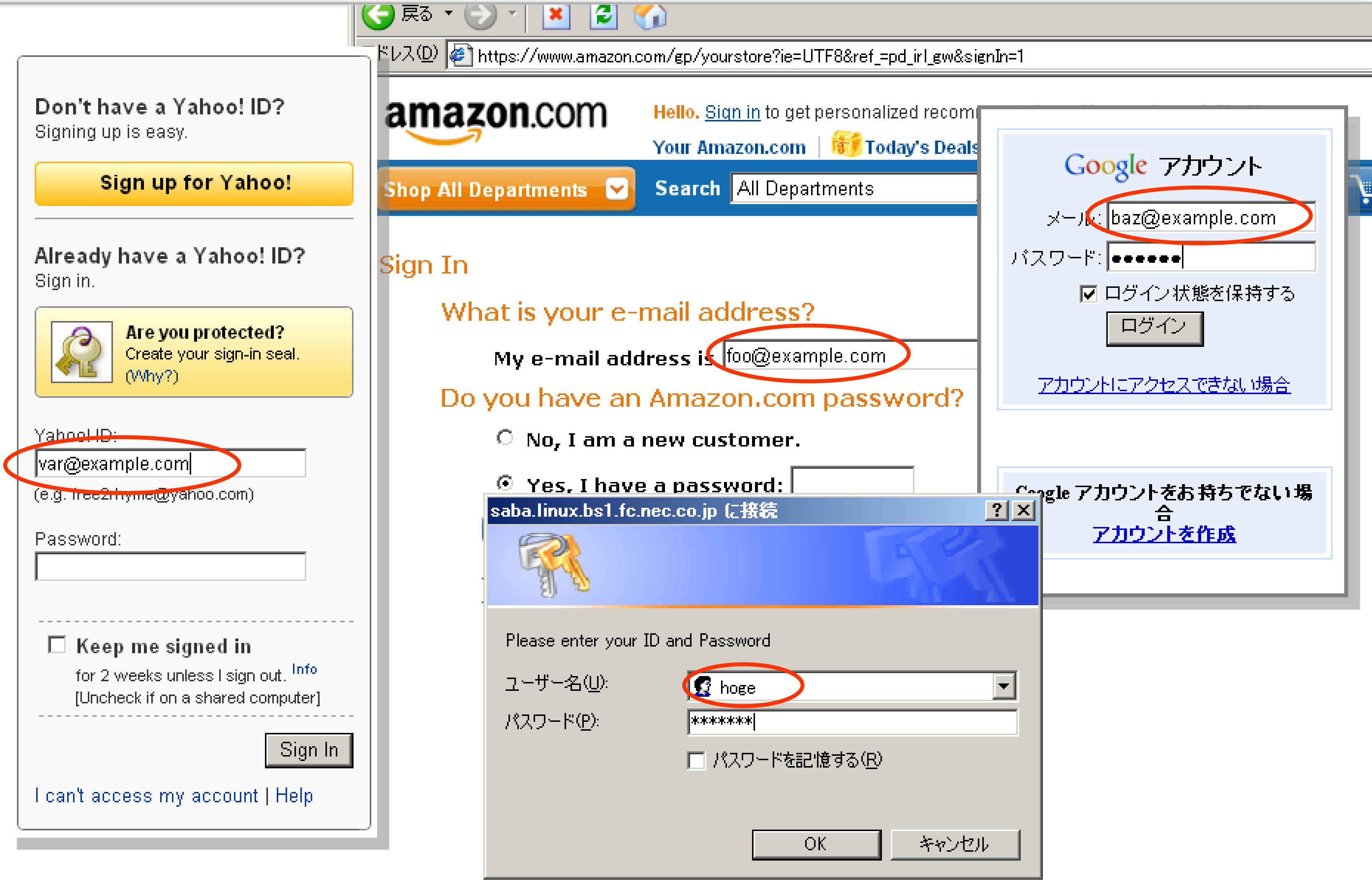
- Apache can check client's Web-ID/PASS (BASIC or DIGEST).
- 403 Error, or Apache launches web-application handlers.

Problem

- Web-application performs with identical privilege of daemon process.
- ➡ It means OS/RDBMS cannot distinguish individual web-users.
- ➡ Web-applications have to work always correctly?

  It means web-applications have to be bugs/vulnerabilities free? ☹

Web users



Don't have a Yahoo! ID?
Signing up is easy.

Sign up for Yahoo!

Already have a Yahoo! ID?
Sign in.

Are you protected?
Create your sign-in seal.
(Why?)

Yahoo! ID: var@example.com
(e.g. free2rhyme@yahoo.com)

Password:

□ Keep me signed in
for 2 weeks unless I sign out. Info
[Uncheck if on a shared computer]

Sign In

I can't access my account | Help

戻る

ドレス(D) https://www.amazon.com/gp/yourstore?ie=UTF8&ref_=pd_irl_gw&signIn=1

amazon.com

Hello. Sign in to get personalized recom

Your Amazon.com | Today's Deals

Shop All Departments | Search | All Departments

Sign In

What is your e-mail address?

My e-mail address is: foo@example.com

Do you have an Amazon.com password?

○ No, I am a new customer.

● Yes, I have a password:

Google アカウント

メール: baz@example.com

パスワード: ●●●●●●

☑ ログイン状態を保持する

ログイン

アカウントにアクセスできない場合

Google アカウントをお持ちでない場合

アカウントを作成

saba.linux.bs1.fc.nec.co.jp に接続

Please enter your ID and Password

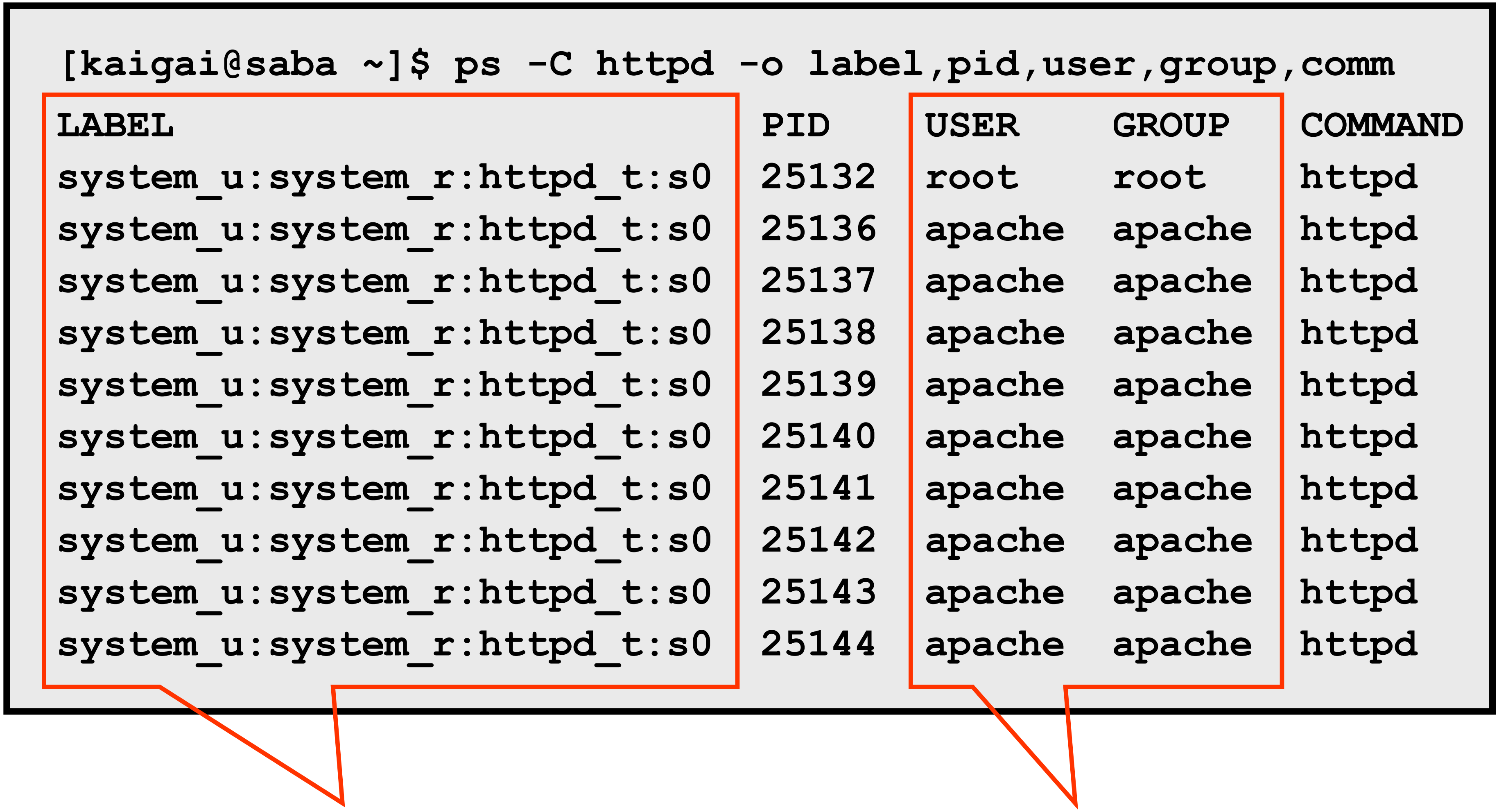
ユーザー名(U): hoge

パスワード(P): *******

□ パスワードを記憶する(R)

OK | キャンセル

Not web-users

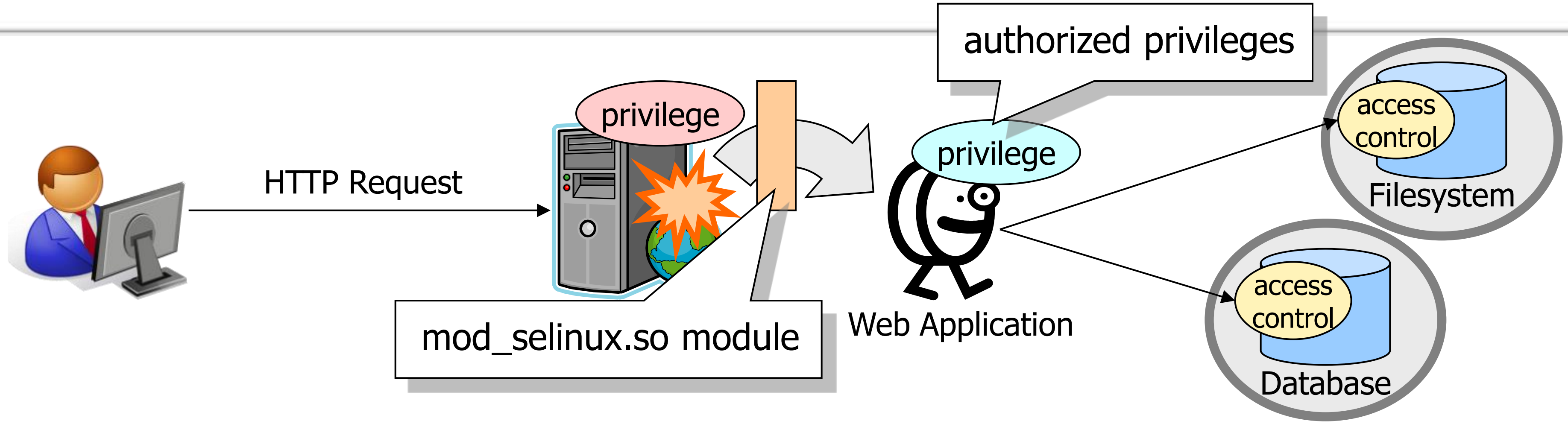


```
[kaigai@saba ~]$ ps -C httpd -o label,pid,user,group,comm
LABEL                          PID    USER     GROUP    COMMAND
system_u:system_r:httpd_t:s0   25132  root     root     httpd
system_u:system_r:httpd_t:s0   25136  apache   apache   httpd
system_u:system_r:httpd_t:s0   25137  apache   apache   httpd
system_u:system_r:httpd_t:s0   25138  apache   apache   httpd
system_u:system_r:httpd_t:s0   25139  apache   apache   httpd
system_u:system_r:httpd_t:s0   25140  apache   apache   httpd
system_u:system_r:httpd_t:s0   25141  apache   apache   httpd
system_u:system_r:httpd_t:s0   25142  apache   apache   httpd
system_u:system_r:httpd_t:s0   25143  apache   apache   httpd
system_u:system_r:httpd_t:s0   25144  apache   apache   httpd
```

Security context of the httpd daemon
used to access controls in SELinux

UNIX Uid/Gid of the httpd daemon
used to discretionary access controls

Apache/SELinux Plus



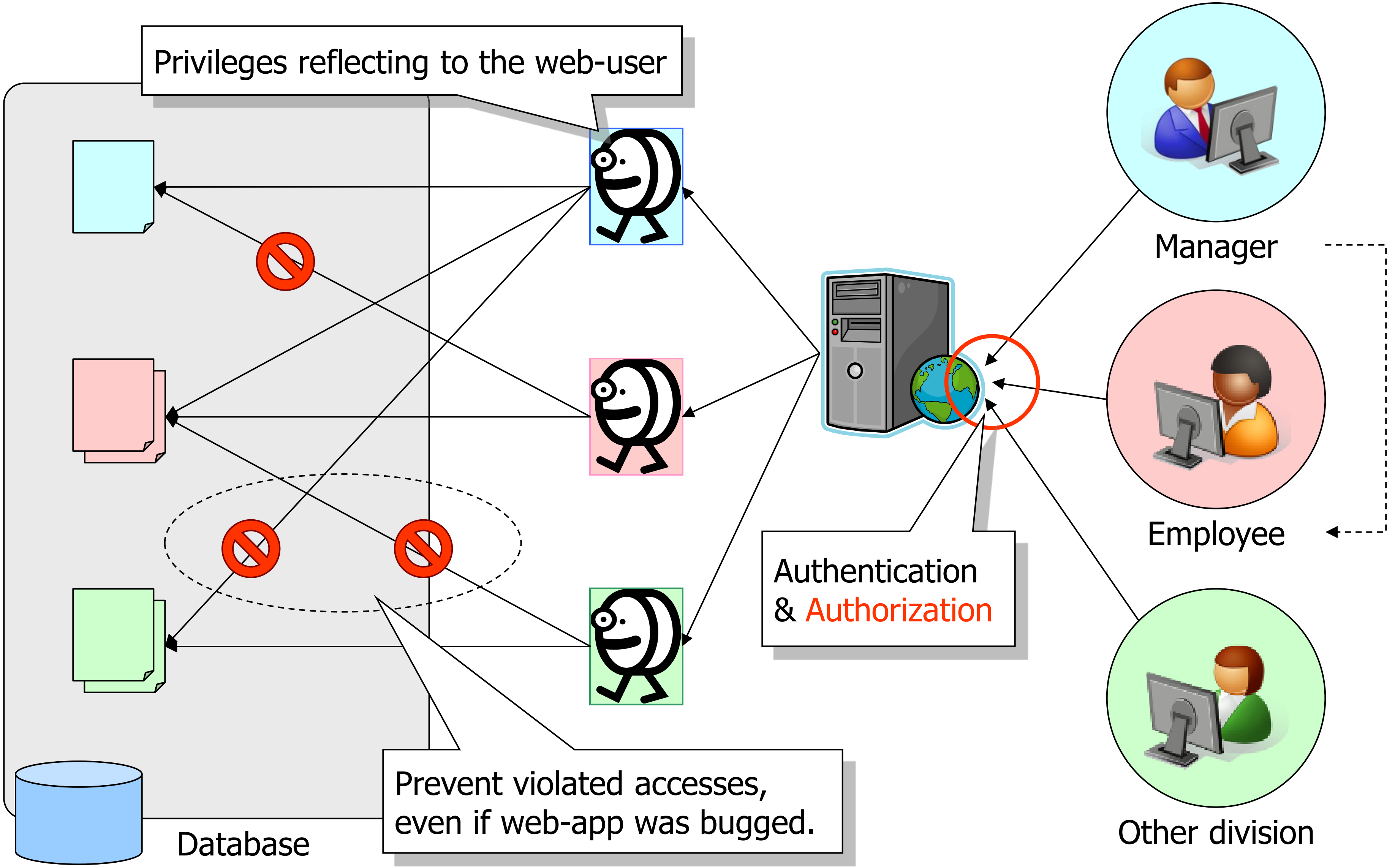
Terms

- Authentication is a function of identifying the connected user.
- Authorization is a function of assigning the rights to resources.

Apache/SELinux Plus (mod_selinux.so)

- It assigns a corresponding security context based on HTTP authentication prior to web-application launches.
- ➡ It enables to confine web-application's accesses.
- Unlike UNIX, no root capabilities are needed to change privileges.

System-image (1/2) : Per web-user separation



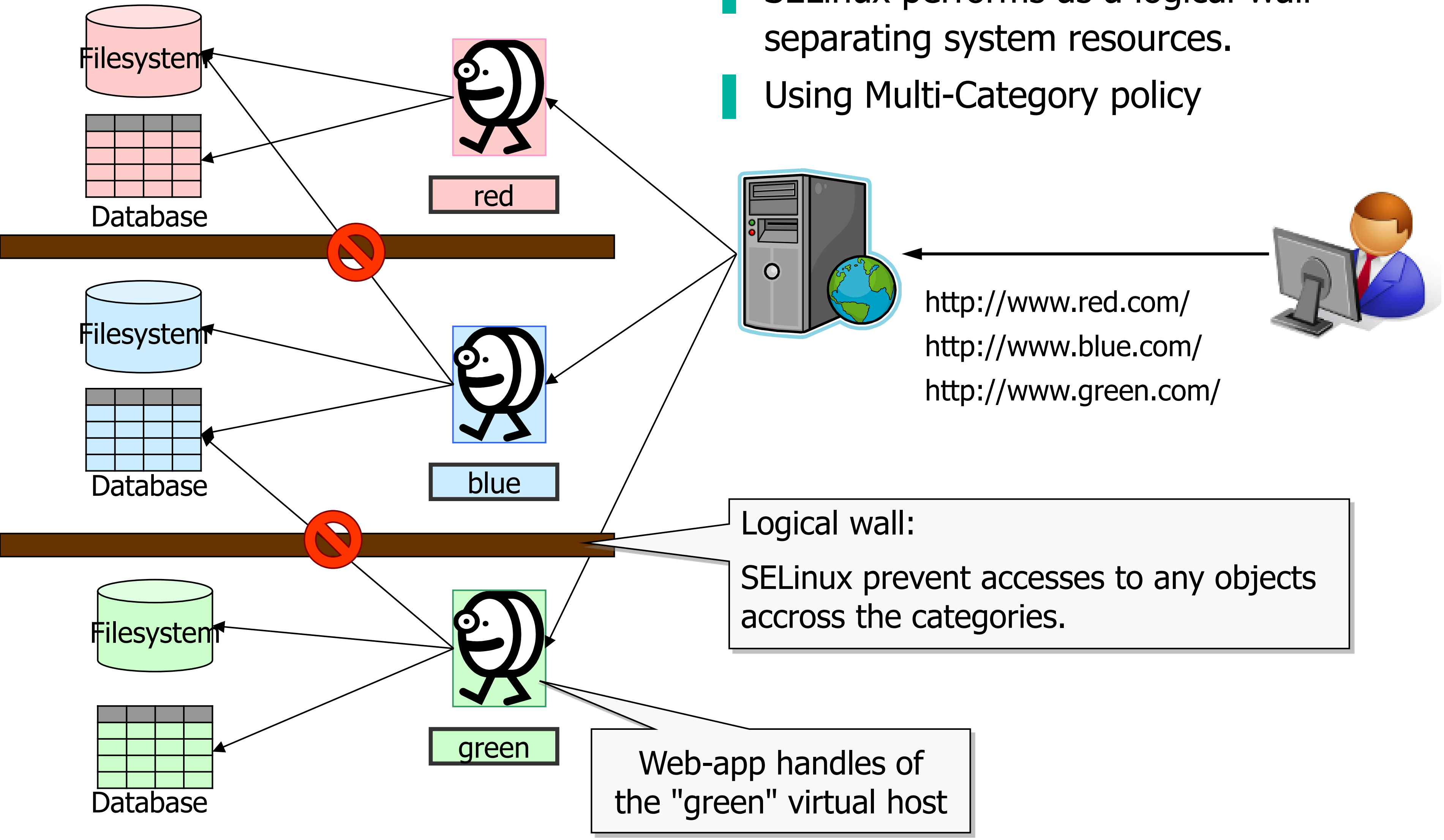
Privileges reflecting to the web-user

Authentication & Authorization

Prevent violated accesses, even if web-app was bugged.

System image (2/2) : Per virtual host separation

- SELinux performs as a logical-wall separating system resources.
- Using Multi-Category policy

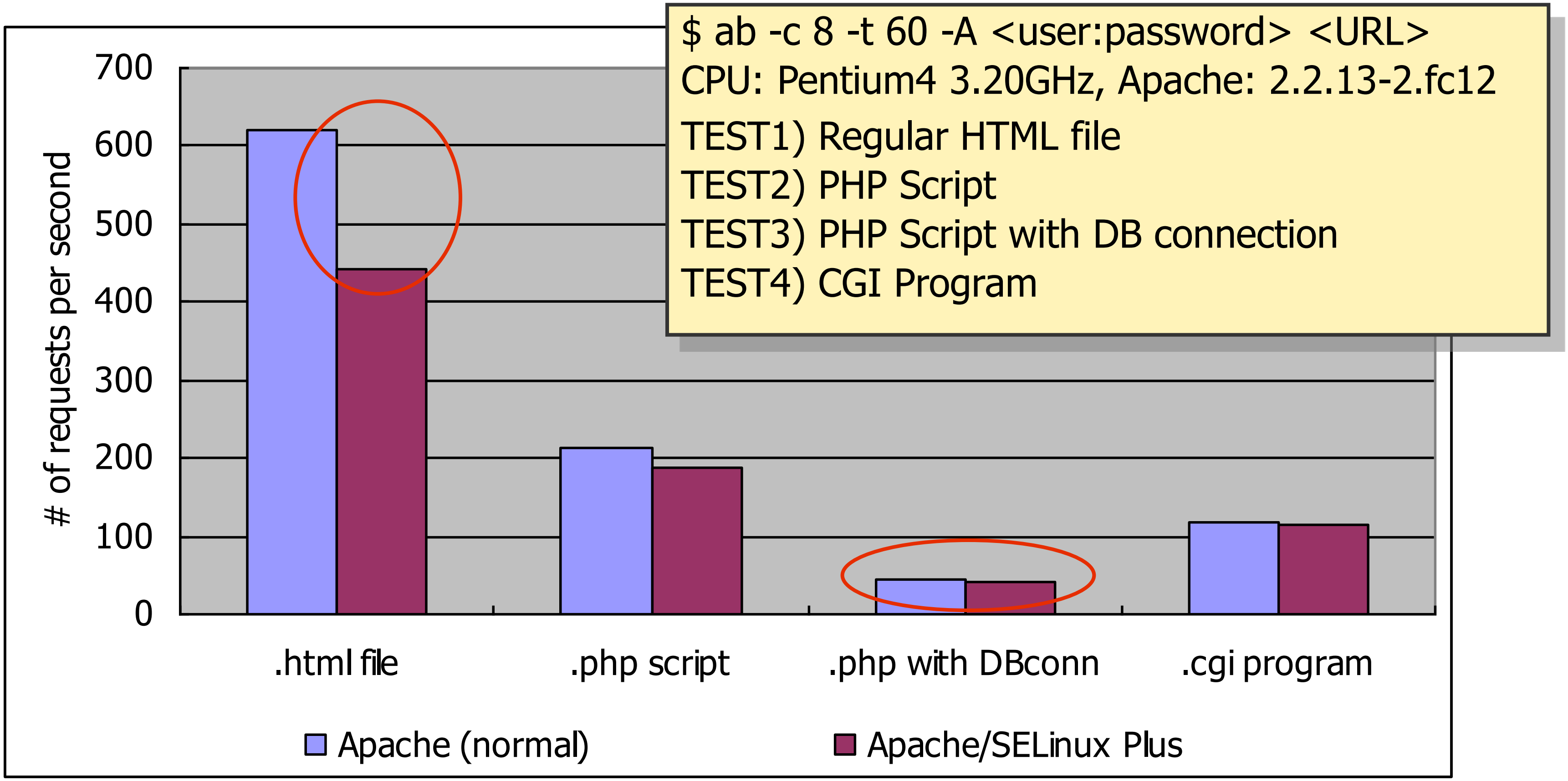


Logical wall:

SELinux prevent accesses to any objects accross the categories.

Web-app handles of the "green" virtual host

Performance - Apache/SELinux Plus



```
$ ab -c 8 -t 60 -A <user:password> <URL>
```

CPU: Pentium4 3.20GHz, Apache: 2.2.13-2.fc12

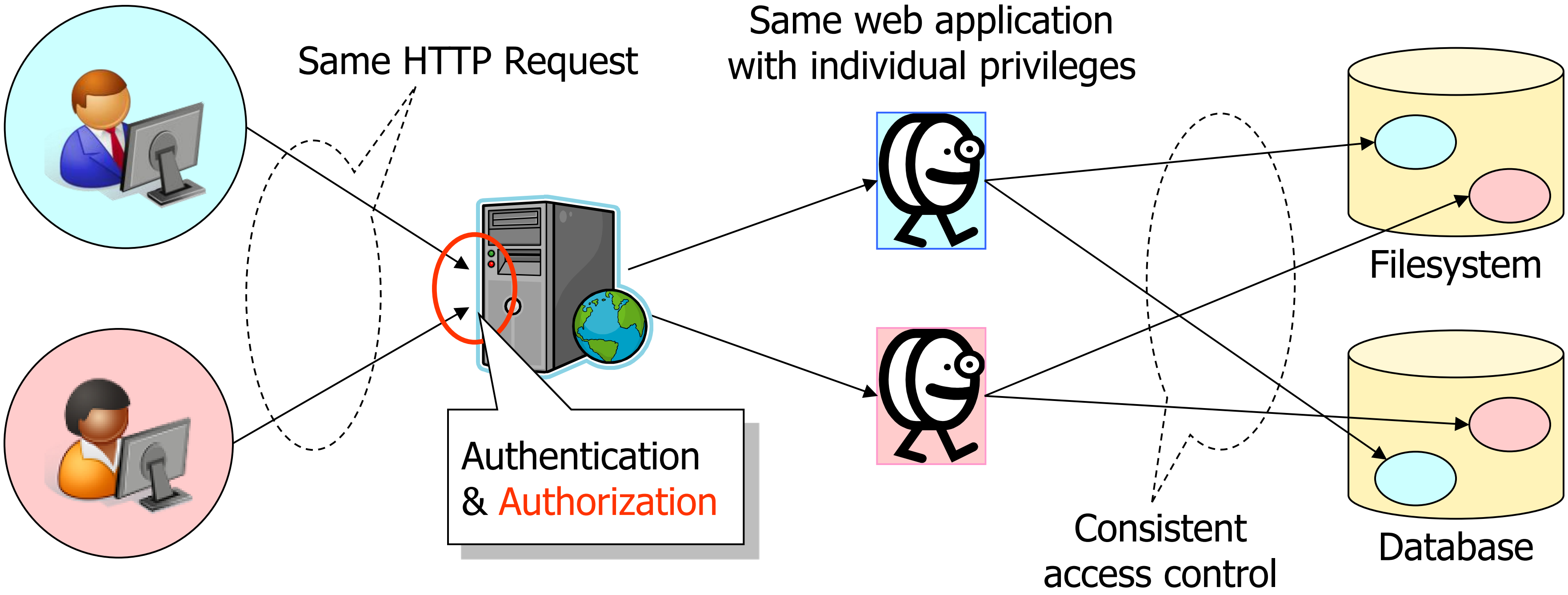
TEST1) Regular HTML file
TEST2) PHP Script
TEST3) PHP Script with DB connection
TEST4) CGI Program

- The cost to assign privileges is relatively large in lightweight request.
- Less differences in our major target (Web+DB applications)
  - Other steps obscures the cost to assign privileges.

Agenda

1. Backgrond
2. SE-PostgreSQL
3. Apache/SELinux plus
4. Demonstration
5. Future Plans

Demonstration



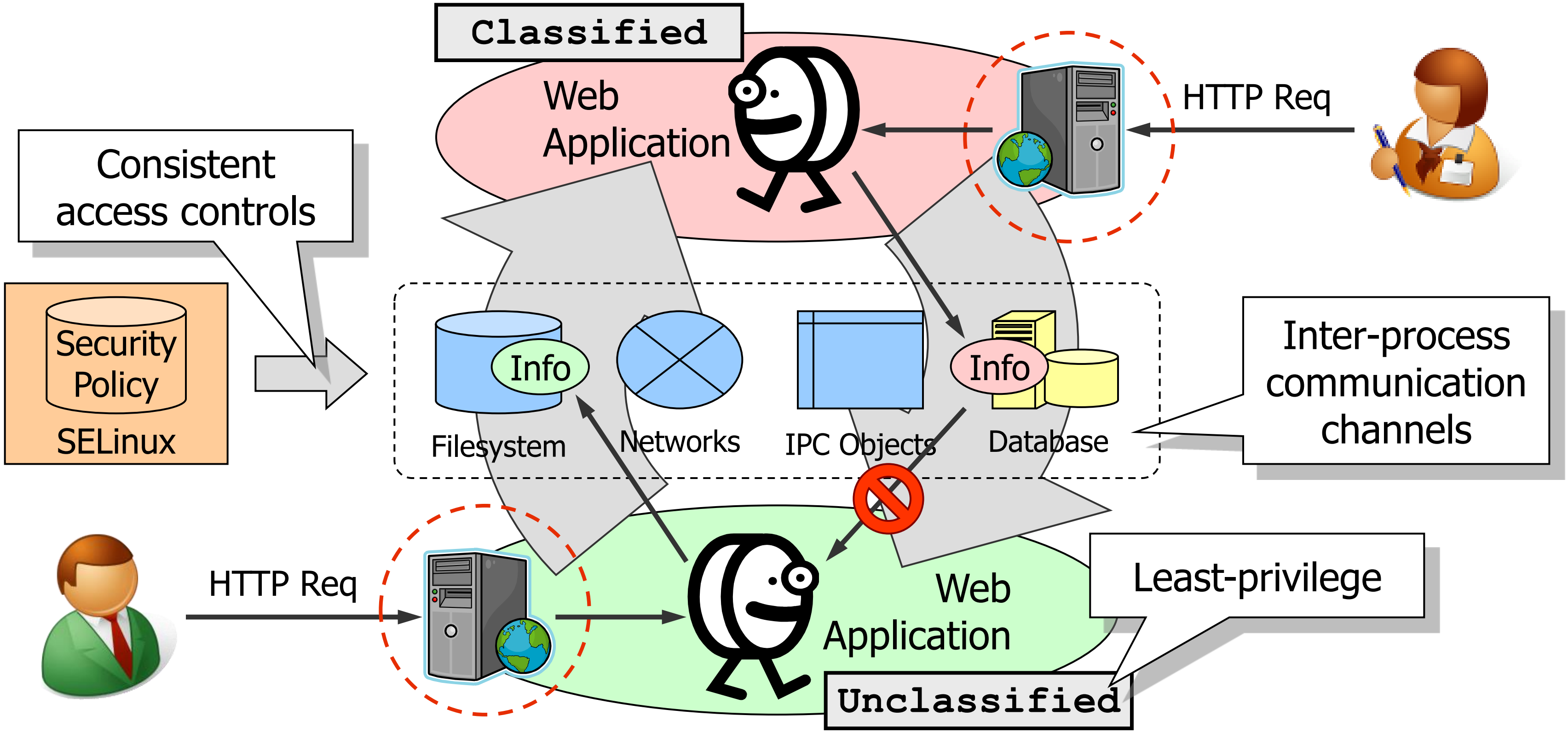
- Apache/SELinux Plus launches a PHP script with individual privileges.
- The PHP script can access both of filesystem and database.
  - Linux applies access controls on filesystems
  - PostgreSQL applies access controls on databases

➡ Consistent access controls, although different mechanisms decide it.

Agenda

1. Backgrond
2. SE-PostgreSQL
3. Apache/SELinux plus
4. Demonstration
5. Future Plans

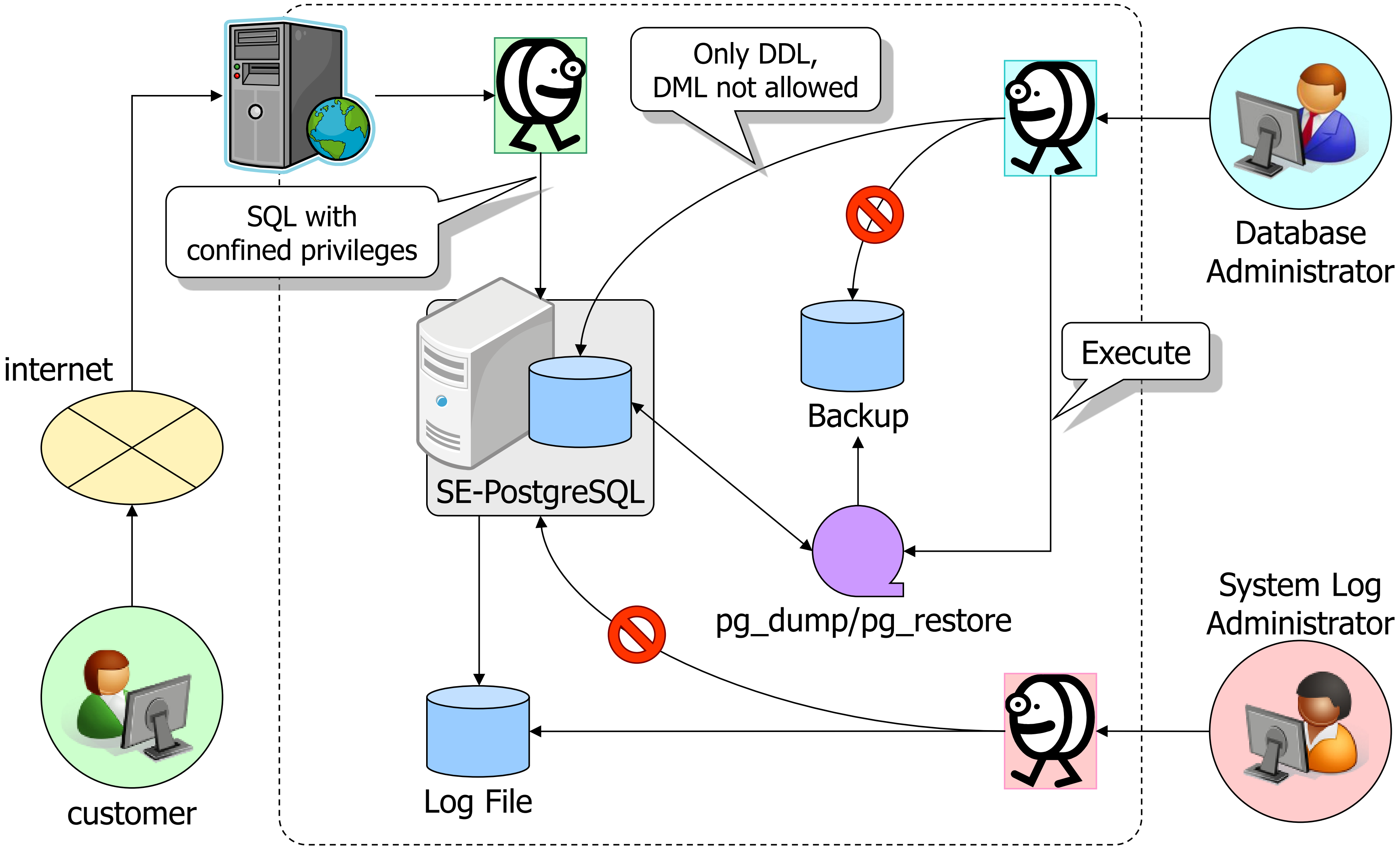
Conceptual diagram of LAPP/SELinux



- SE-PostgreSQL ensures system-wide **consistency** in access controls.
- Apache/SELinux Plus ensures **least-privilege** on web-applications.

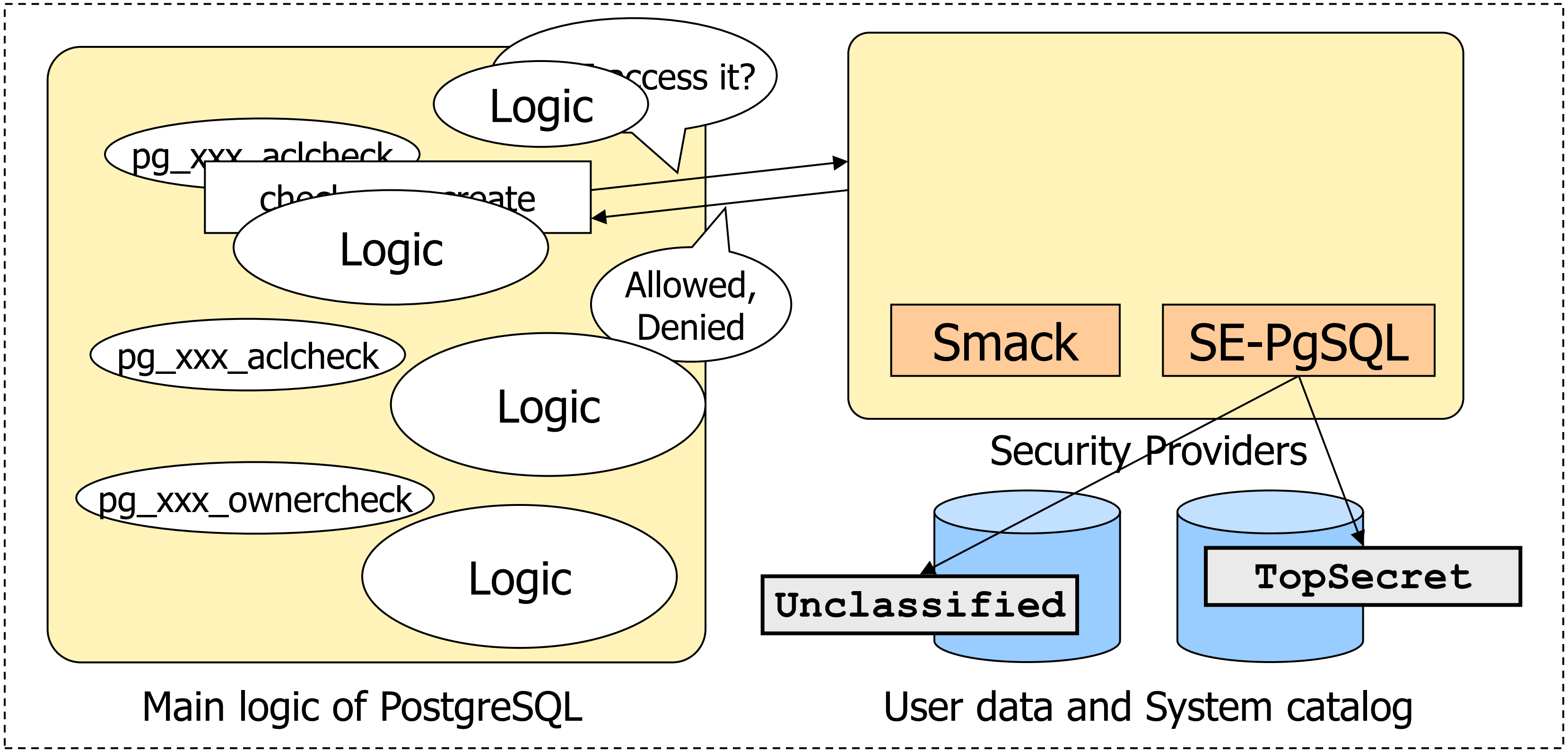
➡ LAPP/SELinux provides a secure web-application platform.

Idea: Role Based Access Control System



Plan to upstream: SE-PostgreSQL

- Access control reworks
- Add security label support
- Add an optional security provider



Summary of LAPP/SELinux

Background

- Web Application's security is Hot issue now.

Key concept

- Utilize SELinux as conductor of access control

Key components

- SE-PostgreSQL
- Apache/SELinux Plus

Road To SE-PostgreSQL being Upstreamed

- External Security Providers
- Security Label Support
- SELinux support; as one of the security providers

➡ Here we go! Let's join us on v9.1 development!

Any Questions?

Thank you!

Empowered by Innovation

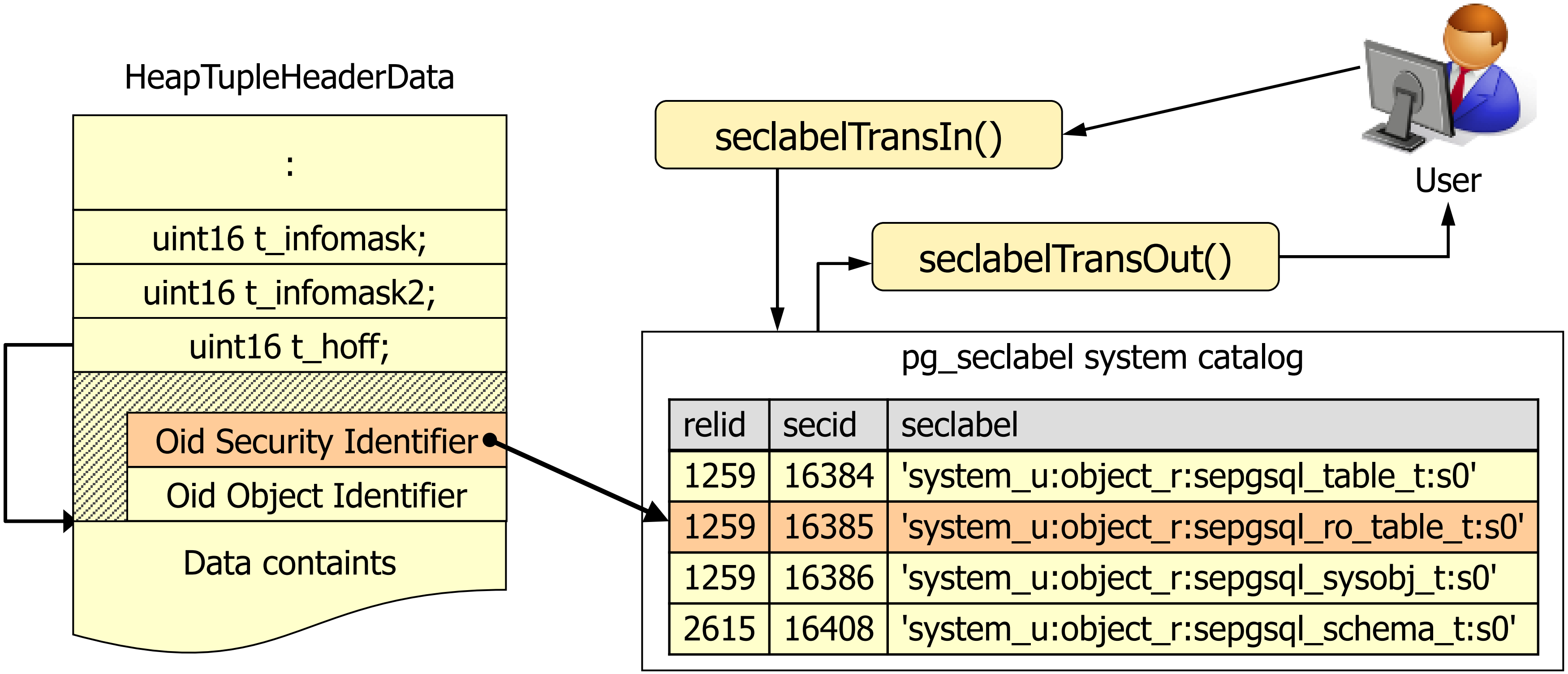
NEC

Appendix

Management of the security labels

Data format

- A tuple has its security context as an object identifier (4-bytes).
  - It minimizes the waste of storage to store security context.
  - It allows to lookup avc cached without text comparison.
- pg_seclabel system catalog holds its text representation.



HeapTupleHeaderData

:

uint16 t_infomask;

uint16 t_infomask2;

uint16 t_hoff;

Oid Security Identifier

Oid Object Identifier

Data containts

seclabelTransIn()

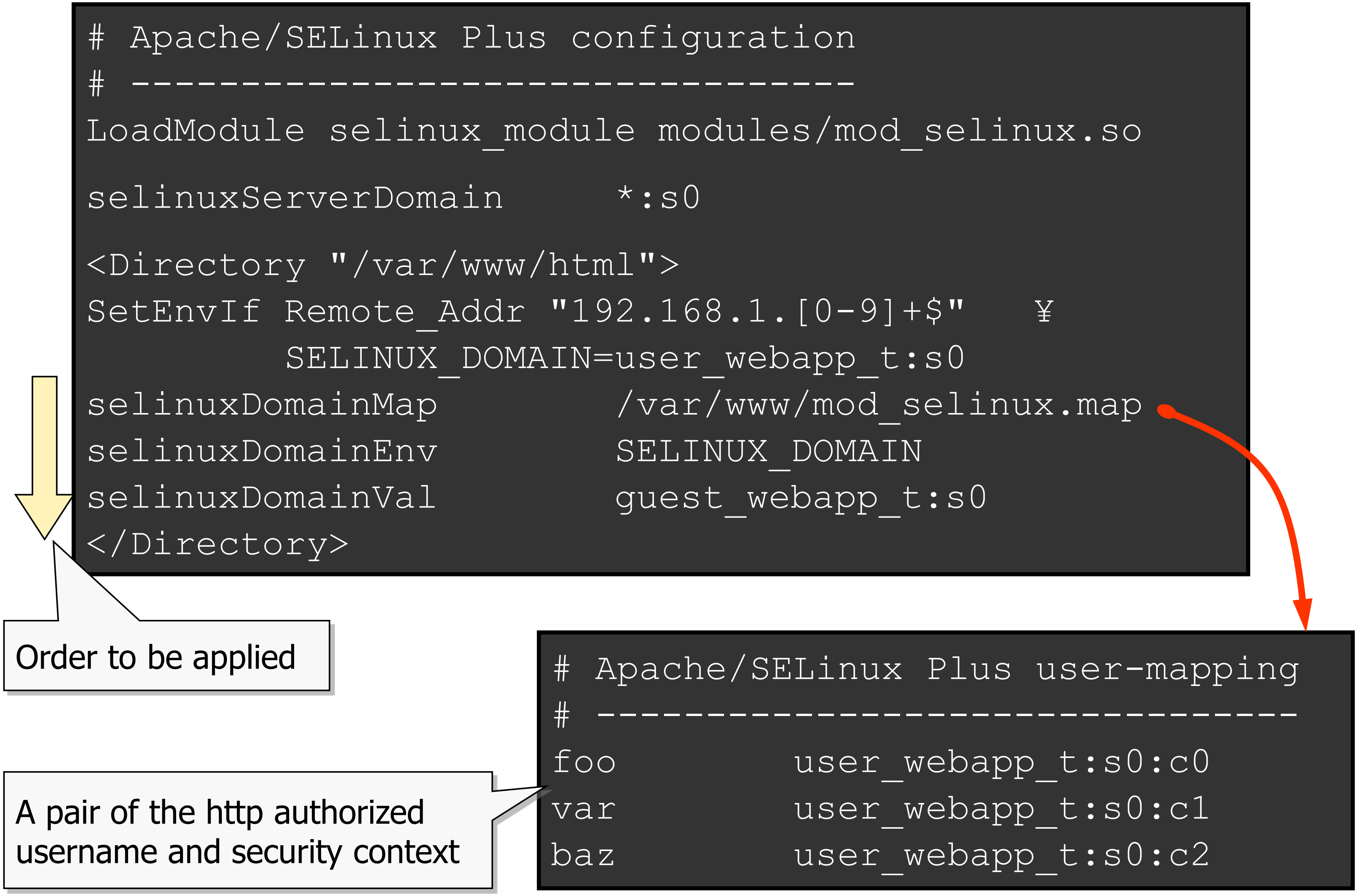
seclabelTransOut()

User

pg_seclabel system catalog

relid	secid	seclabel
1259	16384	'system_u:object_r:sepgsql_table_t:s0'
1259	16385	'system_u:object_r:sepgsql_ro_table_t:s0'
1259	16386	'system_u:object_r:sepgsql_sysobj_t:s0'
2615	16408	'system_u:object_r:sepgsql_schema_t:s0'

Apache/SELinux Plus configuration (1/2)



```
# Apache/SELinux Plus configuration
# -----------------------------------
LoadModule selinux_module modules/mod_selinux.so

selinuxServerDomain     *:s0

<Directory "/var/www/html">
SetEnvIf Remote_Addr "192.168.1.[0-9]+$"   ¥
         SELINUX_DOMAIN=user_webapp_t:s0
selinuxDomainMap        /var/www/mod_selinux.map
selinuxDomainEnv        SELINUX_DOMAIN
selinuxDomainVal        guest_webapp_t:s0
</Directory>
```

Order to be applied

```
# Apache/SELinux Plus user-mapping
# -----------------------------------
foo          user_webapp_t:s0:c0
var          user_webapp_t:s0:c1
baz          user_webapp_t:s0:c2
```

A pair of the http authorized username and security context

Apache/SELinux Plus configuration (2/2)

```
# Apache/SELinux Plus (Per VirtualHost Separation)
# ----------------------------------------
LoadModule selinux_module modules/mod_selinux.so

selinuxServerDomain     *:s0-s0:c0.c1

<VirtualHost *:80>
DocumentRoot            /var/www/html
ServerName              red.example.com
selinuxDomainVal        *:s0:c0
</VirtualHost>

<VirtualHost *:80>
DocumentRoot            /var/www/html
ServerName              blue.example.com
selinuxDomainVal        *:s0:c1
</VirtualHost>
```

Web-server process MUST dominate all the categories.

It assigns c1 category for all the HTTP requests including anonymous ones.