Firebird 3: implementing safe authorization infrastructure

Alex Peshkov

Firebird Foundation
IbPhoenix
2016
Firebird 3: implementing safe authorization infrastructure

- Main authorization-related features:
  - Multiple security databases
  - Authorization plugins
Firebird 3: implementing safe authorization infrastructure

Databases:

- Red
- Green
- Blue
- Pink
- Yellow
Firebird 3: implementing safe authorization infrastructure

Databases:

- Red
- Green
- Blue
- Pink
- Yellow

Group of related DBs
Firebird 3: implementing safe authorization infrastructure

Databases:

Red → RGB → Blue

Group of related DBs

Pink → Yellow

Firebird 3: implementing safe authorization infrastructure

databases.conf (former aliases.conf)

Red=/db/Red.fdb
{
    SecurityDatabase=RGB
}
Green=/db/Green.fdb
{
    SecurityDatabase=RGB
}
Blue=/db/Blue.fdb
{
    SecurityDatabase=RGB
}
Pink=/db/Pink.fdb
{
    SecurityDatabase=Pink
}
Yellow=/reserve/Yellow.fdb
{
    SecurityDatabase=Yellow
}
RGB=/reserve/Magenta.fdb
Firebird 3: implementing safe authorization infrastructure

- What about cross-database query?

EXECUTE STATEMENT SQL_TEXT
ON EXTERNAL 'Green'

- Only between databases with same security database
Firebird 3: implementing safe authorization infrastructure

Databases:

- Red
- Green
- Blue
- Pink
- Yellow
Firebird 3: implementing safe authorization infrastructure

- What about cross-database query?

```sql
EXECUTE STATEMENT SQL_TEXT
ON EXTERNAL 'Yellow'
USER 'Login' PASSWORD '***'
```

- Metadata (procedure sources) is world readable
Firebird 3: implementing safe authorization infrastructure

- What about cross-database query?

```sql
EXECUTE STATEMENT SQL_TEXT
ON EXTERNAL 'Yellow'
USER 'Login' PASSWORD '***'
```

- INSECURE!!!
Firebird 3: implementing safe authorization infrastructure
Firebird 3: implementing safe authorization infrastructure

- CREATE MAPPING
  - Authorization source
  - Existing authorization object
  - New authorization object

- CREATE MAPPING <name>
  USING clause
  FROM clause
  TO clause
Firebird 3: implementing safe authorization infrastructure

- **USING clause:**
  - USING PLUGIN <name> [ IN <sec-db name> ]
    - using plugin srp
    - using plugin legacy_auth in pink
    - using plugin win_sspi in yellow
  - USING ANY PLUGIN [ IN <sec-db name> ]
    - using any plugin in pink
  - USING ANY PLUGIN SERVERWIDE
  - USING MAPPING [ IN <sec-db name> ]
    - using mapping in pink
  - USING * [ IN <sec-db name> ]
    - using * in rgb
Firebird 3: implementing safe authorization infrastructure

Databases:

- Red
- Green
- Blue
- Pink
- Yellow

CREATE MAPPING FROM_RGB USING * IN RGB ...

RGB

Databases:
Firebird 3: implementing safe authorization infrastructure

- FROM clause
  - FROM <object-type> <object-name>
    - from user sysdba
    - from group administrators
  - FROM ANY <object-type>
    - from any user
CREATE MAPPING FROM_RGB USING * IN RGB FROM ANY USER ...
Firebird 3: implementing safe authorization infrastructure

- TO clause
  - TO ROLE [ <role-name> ]
    - to role
  - TO USER [ <user-name> ]
    - to user RedGuest
CREATE MAPPING FROM_RGB USING * IN RGB
FROM ANY USER TO USER RGBGUEST
Firebird 3: implementing safe authorization infrastructure

- Using services API with multiple security databases
  
  - `fbsvcmgr localhost:service_mgr
    user sysdba password masterkey
    action_db_stats dbname Yellow`

  - `fbsvcmgr localhost:service_mgr
    user sysdba password YellowMaster
    action_db_stats dbname Yellow`

- Both fail
Firebird 3: implementing safe authorization infrastructure

Using services API with multiple security databases

Services manager

fbsvcmgr localhost:service_mgr
user sysdba password masterkey
action_db_stats dbname Yellow
Firebird 3: implementing safe authorization infrastructure

Using services API with multiple security databases

Services manager

```
fbsvcmgr localhost:service_mgr
user sysdba password YellowMaster
action_db_stats dbname Yellow
```
Firebird 3: implementing safe authorization infrastructure

- Using services API with multiple security databases
  - New parameter when attaching to server: `isc_spb_expected_db <dbname>`
  - `fbsvcmgr localhost:service_mgr user sysdba password YellowMaster expected_db Yellow action_db_stats dbname Yellow`
Firebird 3: implementing safe authorization infrastructure

Using services API with multiple security databases

RGB

Security.db

Services manager
Firebird 3: implementing safe authorization infrastructure

- Using services API with multiple security databases
  - Create appropriate mapping:
    - create mapping DefDb
      using plugin Srp in "security.db"
      from user sysdba to user
    - fbsvcmgr localhost:service_mgr
      user sysdba password masterkey
      action_db_stats dbname Yellow
CREATE MAPPING DEFDBA USING PLUGIN SRP IN "SECURITY.DB" FROM USER SYSDBA TO USER

Using services API with multiple security databases
Firebird 3: implementing safe authorization infrastructure

- Using trace API with multiple security databases
  - Audit session can access any database
  - Use expected_db (like any other service)
  - Use mapping

- Use of mapping makes it possible to trace databases from different security groups in same trace session
Firebird 3: implementing safe authorization infrastructure

- SRP
  - Resistant to all kinds of attacks except brute force
  - Produces unique strong session key (needed for wire encryption)

- Why we need other plugins?
Firebird 3: implementing safe authorization infrastructure

- Plugin implementing “trusted computer”
  - Each login from given box should have fixed predefined user name (CURRENT_USER)
  - No need to enter login/password

- Environment
  ISC_USER=BigBoss
  ISC_PASSWORD=BossPassword

- Windows trusted authentication
  - OS dependent
  - No encryption key
Firebird 3: implementing safe authorization infrastructure

- OS independent plugin (RsaPair)

  - Pass pre-configured login
  - Decrypt random value and pass it's hash
  - Pick public key for login, encrypt random value and pass it
  - Compare passed hash
  - Authorization complete
Firebird 3: implementing safe authorization infrastructure

- OS independent plugin with encryption

1. Pass pre-configured login
2. Decrypt random value and pass it's hash
3. Pick public key for login, encrypt random value and pass it
4. Compare passed hash
5. Authorization complete

Use random value for wire encryption
Firebird 3: implementing safe authorization infrastructure

- Use configuration file (RsaPair.conf) for setup

- Server side:
  BigBoss=<boss public key (long hex string)>
  Management=<one more public key>

- Client side:
  Login=BigBoss
  Key=<boss private key>
CREATE GLOBAL MAPPING BOSSSES USING PLUGIN RsaPair FROM ANY USER TO USER;
Firebird 3: implementing safe authorization infrastructure

- OS independent plugin with encryption

- Use random value for wire encryption
- Pick public key for login, encrypt random value and pass it
- Compare passed hash
- Authorization complete
- Set trusted role
- Use random value for wire encryption
- Decrypt random value and pass its hash
- Pass pre-configured login
CREATE GLOBAL MAPPING RsaRoles
USING PLUGIN RsaPair
FROM ANY ROLE TO ROLE;

- RsaPair.conf:
  BigBoss=<boss public key>
  BigBossRole=GuestRole
  Management=<one more public key>
Thanks for your attention!