Overview of Red Database 2.5

Roman Simakov, director of system development department

RED SOFT CORPORATION
About company

- Red Soft Corporation was founded in 2006
- All solutions based on Open Source code
- The main product is Red Database
Development process (schema)
QA process

Repository server

Commit changes

Developer

Pull changes and run build jobs

Jenkins server

Publish builds

Snapshots
Release Candidates
Releases

Test manager

View test result
Security features

- Cryptographic plugin
- Multi-factor authentication
- Cumulative roles
- DML access control
- DDL access control
- Service access control
- Record filtering

Functional features

- Java Stored Procedures
- Full Text Search
- OpenLDAP integration
- StandBy cluster (engine-level replication)
Cryptographic plugin

• It's a key feature used in other features to perform cryptographic operations.

• It allows users to use necessary cryptographic methods required in different countries by using related plugins.

• For Russia CryptoPro library and Windows CryptoAPI are supported.
Multi-factor authentication

• It allows user to provide several factors to be authenticated: OS context, password, certificate, etc.

• Access to database is defined by login policy. It says what factors user must provide for authentication.

• While authentication all authentication factors are transferred in encoded form.

• After authentication both client and server have session key for exchanging private messages, for example new password when user wants to change it.
## Login policy properties

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTH_FACTORS</td>
<td>Example: (WINDOWS_NTLM PASSWORD)</td>
</tr>
<tr>
<td>PSWD_NEED_CHAR</td>
<td>The minimum number of characters in the password</td>
</tr>
<tr>
<td>PSWD_NEED_DIGIT</td>
<td>The minimum number of digits in the password</td>
</tr>
<tr>
<td>PSWD_NEED_DIFF_CASE</td>
<td>Need to use different case of characters in the password</td>
</tr>
<tr>
<td>PSWD_MIN_LEN</td>
<td>The minimum password length</td>
</tr>
<tr>
<td>PSWD_VALID_DAYS</td>
<td>The password validation interval in days</td>
</tr>
<tr>
<td>PSWD_UNIQUE_COUNT</td>
<td>The minimum number of the last unique passwords</td>
</tr>
<tr>
<td>MAX_FAILED_COUNT</td>
<td>The maximum number of failed attempt of authentication</td>
</tr>
<tr>
<td>MAX_SESSIONS</td>
<td>The maximum number of user sessions to database server</td>
</tr>
<tr>
<td>MAX_IDLE_TIME</td>
<td>The maximum idle time interval to user disconnecting</td>
</tr>
</tbody>
</table>
Login policies

DDL commands to control policies

```
CREATE POLICY <policy_name> AS [param = value [, param = value]];  
DROP POLICY <policy_name>;  
ALTER POLICY <policy_name> AS [param = value [, param = value]];  
```

To grant policy to user use

```
GRANT POLICY <policy_name> TO <user_name>;  
```

To revoke policy from user just grant DEFAULT policy to him

```
GRANT POLICY "DEFAULT" TO <user_name>;  
```
Cumulative roles

You can grant role to role except circle references

```
GRANT ROLE1 TO ROLE2;
REVOKE ROLE1 FROM ROLE2;
```

- if user doesn't specify a role he gets permissions of all roles granted to him;
- if user specifies a role he takes privileges of this role only.
DML access control

Extended permissions for generators/sequences

```
GRANT SELECT | ALTER ON GENERATOR <generator> TO {<user> | <role>} [WITH GRANT OPTION];

REVOKE SELECT | ALTER ON GENERATOR <generator> FROM {<user> | <role>};
REVOKE GRANT OPTION FOR SET | GET ON GENERATOR <generator> FROM {<user> | <role>};
```

Extended permissions for table columns

```
GRANT SELECT | INSERT | UPDATE { column [, ... ] } ON [TABLE] <table> TO {<user> | <role>} [WITH GRANT OPTION]

REVOKE SELECT | INSERT | UPDATE { column [, ... ] } ON [TABLE] <table>
          FROM {<user> | <role>}

REVOKE GRANT OPTION FOR SELECT | INSERT | UPDATE { column [, ... ] } ON [TABLE] <table> FROM {<user> | <role>}
```
DDL access control (now in Firebird 3 too)

Extended permissions for creating objects of database

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRANT CREATE OBJECT TO {&lt;USER&gt;</td>
</tr>
<tr>
<td>REVOKE CREATE OBJECT FROM {&lt;USER&gt;</td>
</tr>
</tbody>
</table>

Extended permissions for altering/dropping objects of database

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRANT ALTER</td>
</tr>
<tr>
<td>REVOKE ALTER</td>
</tr>
</tbody>
</table>

Where OBJECT can be:

- TABLE, TRIGGER, PROCEDURE, VIEW, DOMAIN, ROLE, GENERATOR,
- SEQUENCE, EXCEPTION, SHADOW, FUNCTION, INDEX, POLICY
Service access control

It's able to grant permissions to start some services

(GBAK, GFIX, GSTAT, GSEC)

GRANT EXECUTE ON SERVICE <SERVICE_NAME> TO {<USER>|<ROLE>}

REVOKE EXECUTE ON SERVICE <SERVICE_NAME> FROM {<USER>|<ROLE>}

• Permissions can be granted to users or global roles stored in security2.fdb.
• Permissions can be granted by SYSDBA or by user with SECADMIN global role.
Record filtering

- Based on special SELECT triggers.
- It allows user to skip the records if the given condition is false.
- It allows user to clear some fields of records if the given condition is false.
- It's used to filter system catalog to prevent user without any permissions on database object even to know about its existence.
Record filter syntax

User can declare filters in CREATE TABLE

```
CREATE TABLE <table_name> [EXTERNAL [FILE] "<filespec>" ] (<col_def> [, <col_def> | <tconstraint> ...], [COLFILTER <col_name> (<condition>), ...]) [, RECFILTER (<condition>)]
```

To manage filters use ALTER TABLE

```
ALTER TABLE table SET RECFILTER (<condition>);
ALTER TABLE table DROP RECFILTER;
ALTER TABLE table SET COLFILTER <col_name> (<condition>);
ALTER TABLE table DROP COLFILTER <col_name>;
```
Java Stored Procedures

• It's possible to develop both user defined procedures and user defined functions
• Portable code on widely used programming language
• It's possible to re-use a lot of libraries
• Java SP can return result set which allow them to be used as data source.
• Can be used to exchange data with other databases.
Full Text Search

- Based on high performance cross-platform engine lucene (https://lucene.apache.org/)
- Can perform search by several tables and fields
- Can search in the most widely used file formats: rtf, doc, open office, pdf, etc.
Integration with OpenLDAP
Optimization work with Storages

- SAS, SSD, ... (fast, expensive)
- HDD (slow, cheap)

Active DB part

Archive DB part
StandBy cluster (with sync replication)

Master
+ pacemaker agent

Record level changes async sent and wait applying before transaction commit

Slave
+ pacemaker agent

HA cluster under Pacemaker*

* More about pacemaker at http://clusterlabs.org
Automated Information System of Federal Service for Officers of Justice of Russia

- AIS is installed and work in 85 regional departments and in the main office of FSOJ of Russia
- Total amount of Red Database installations are about 2720, i.e. every city of Russia has one or several Red Database servers
- AIS handles more than $10^9$ documents per year
- AIS works in 24/7 mode
- Some databases more than 1TB and a lot of data goes to archived set of database files
- 100x of concurrent connections
- 100 000x transactions per hour
Regional medical information system

- Partner is SmartDeltaSystems Ltd. ([http://www.sdsys.ru/](http://www.sdsys.ru/))
- Migration from Firebird because of they need to have certified solution and support
- Work on CentOS and Windows
- ~200 installations
- Size of databases up to 12 GB
- Central database size is about 50 GB
- 1000x concurrent connections
Automated server of radiomonitoring of Russia

- The main database is ~700 GB
- Regional - 100x GB
- Increased by 10x GB per year
- 600 000 transactions per day
- 100x concurrent connections
- OpenLDAP authentication
Awards

• «The best project of the state sector – 2011» (http://www.raspo.ru/content/28.html)
• AIS FSOJ of Russia was endorsed by Prime Minister of Russia in 2014 (http://government.ru/news/10513)
Some words about Red Database 2.6

- Direction to “state secret” security level
- Mandatory access control based on SELinux integration
- Full database encryption
- Column data encryption by user key
- Traffic and backup files encryption
- Still based on Firebird 2.5
SELinux

libselinux

Policies
packages

Database
objects
initial contexts file

Read during database creation

MAC plugins interface

SELinux plugin

DDL
DML

Red Database

Objects contexts

Users contexts

security2.fdb

Database
user_a (rdb_user_u:rdb_user_r:rdb_user_t:s1)

select * from A
0 records filtered

DB.fdb (system_u:object_r:rdb_database_t:s0)
A (system_u:object_r:rdb_table_t:s0)

<table>
<thead>
<tr>
<th>DATA (system_u:object_r:rdb_column_t:s0)</th>
<th>MAC$LABEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>foo</td>
<td>system_u:object_r:rdb_record_t:s0</td>
</tr>
<tr>
<td>bar</td>
<td>system_u:object_r:rdb_record_t:s1</td>
</tr>
</tbody>
</table>

select * from A
1 record filtered

user_b (rdb_user_u:rdb_user_r:rdb_user_t:s0)
Database Encryption

Key management
CREATE KEY <key name> <algorithm id>
GRANT KEY <key name> TO <user name>
REVOKE KEY <key name> FROM <user name>
DROP KEY <key name>

Full database encryption
isql -mf -certificate <cert alias> [-en(crypt) <key name>]
SQL> CREATE DATABASE <db name>;

Column database encryption
isql -mf -certificate <cert alias>
SQL> CREATE TABLE <table name> (<column def> [, ENCRYPT <column name> USING <key name>]);
SQL> ALTER TABLE <table name> ENCRYPT <column name> USING <key name>;
SQL> ALTER TABLE <table name> DECRYPT <column name>;

Create an encrypted backup
gbak [-en(crypt) <key name>]
Some big goals of Red Database 3.0

- Merge with Firebird 3.0
- Load balancing cluster
- Parallel backup/restore
- GUI tool which support all Red Database features
- Support of OpenGIS specification
- Tools for migration from other DBMSs
Thanks!

We are pleased to invite you to test Red Database!

visit: www.red-soft.biz
ask: rdb.support@red-soft.biz
roman.simakov@red-soft.biz