

Configurez ODBC sur ISE 2.1 avec PostgreSQL

Contenu

[Introduction](#)

[Conditions préalables](#)

[Conditions requises](#)

[Composants utilisés](#)

[Configurez](#)

[Étape 1. Configuration de base de PostgreSQL](#)

[Étape 2. Configuration ISE](#)

[Étape 3. Configurez l'authentification de l'utilisateur](#)

[Étape 4. Configurez la récupération de groupe](#)

[Étape 5. Configurez la récupération d'attributs](#)

[Vérifiez](#)

[Dépannez](#)

[Références](#)

Introduction

Ce document décrit comment configurer le Cisco Identity Services Engine (ISE) avec le serveur de PostgreSQL pour l'authentification ISE utilisant la Connectivité de base de données ouverte (ODBC).

Remarque: L'authentification de la Connectivité de base de données ouverte (ODBC) exige d'ISE de pouvoir chercher un mot de passe utilisateur de texte brut. Le mot de passe peut être chiffré dans la base de données, mais doit être déchiffré par la procédure stockée.

Conditions préalables

Conditions requises

Cisco vous recommande de prendre connaissance des rubriques suivantes :

- Logiciel Cisco Identity Services Engine 2.1
- Base de données et concepts ODBC
- PostgreSQL

[Composants utilisés](#)

Les informations contenues dans ce document sont basées sur les versions de matériel et de logiciel suivantes :

- Cisco Identity Services Engine 2.1
- Centos 7
- PostgreSQL 9.2

Configurez

Remarque: Code du festin SQL dans ce document comme exemple. Habituellement il y a plus d'une manière de coder a désiré la fonctionnalité et tous ont leurs avantages et inconvénients.

Étape 1. Configuration de base de PostgreSQL

Les étapes de configuration incluent la création de base de données et un utilisateur pour ISE avec des autorisations d'accéder à cette base de données.

1. De l'utilisateur de postgres créez l'utilisateur d'isedb :

```
$ createuser --interactive
Enter name of role to add: isedb
Shall the new role be a superuser? (y/n) n
Shall the new role be allowed to create databases? (y/n) y
Shall the new role be allowed to create more new roles? (y/n) n
Password:
```

2. Créez une base de données

```
$ createdb isedb
```

ou avec le SQL :

```
CREATE DATABASE isedb WITH TEMPLATE = template0 OWNER = isedb;
REVOKE ALL ON DATABASE isedb FROM PUBLIC;
REVOKE ALL ON DATABASE isedb FROM postgres;
GRANT CONNECT,TEMPORARY ON DATABASE isedb TO PUBLIC;
GRANT ALL ON DATABASE isedb TO isedb;
```

3. Permettez l'accès à PostgreSQL

```
sudo vi /var/lib/pgsql/data/pg_hba.conf
```

Trouvez les lignes qui ressemble à ceci, près du bas du fichier :

```
host all all 127.0.0.1/32 ident
host all all ::1/128 ident
```

Remplacez alors l'**ident** par le **MD5**, ainsi ils ressemblent à ceci :

```
host all all 127.0.0.1/32 md5
host all all 10.0.0.0/8 md5
```

4. Permettez les connexions distantes à PgSQL

Vous devez ouvrir le fichier de configuration `/var/lib/pgsql/data/postgresql.conf` de PostgreSQL. Ligne de configuration de découverte qui lit :

```
listen_addresses='localhost'
```

et modification à

```
listen_addresses='*'
```

Permettez les connexions de toutes les adresses. Ligne de configuration des ports d'Uncomment (si commenté) :

```
port = 5432
```

5. Reprise PgSQL :

```
$ sudo systemctl start postgresql  
$ sudo systemctl enable postgresql
```

Étape 2. Configuration ISE

Créez une source d'identité ODBC à la **gestion > source extérieure d'identité > ODBC** et connexion de test :

[ODBC List > pgSQL](#)

ODBC Identity Source

General **Connection** Stored Procedures Attributes Groups

ODBC DB connection details

* Hostname/IP[:port]

* Database name

Admin username ⓘ

Admin password

* Timeout

* Retries

* Database type

Test connection X

Connection succeeded

Stored Procedures

- Plain text password authentication - Not Configured
- Plain text password fetching - Not Configured
- Check username or machine exists - Not Configured
- Fetch groups - Not Configured
- Fetch attributes - Not Configured

Étape 3. Configurez l'authentification de l'utilisateur

L'authentification ISE à ODBC utilise le stored procedures. Il est possible au type de sélection de procédures. Dans cet exemple nous utilisons des paramètres en tant que retour. Pour d'autres procédures, référez-vous au [guide d'administration du Logiciel Cisco Identity Services Engine 2.1](#)

Conseil : Il est possible de renvoyer des paramètres Désignés au lieu du resultset. C'est juste un type différent de sortie, fonctionnalité est identique.

1. Créez la table. Assurez-vous que vous avez placé les configurations d'identité sur la clé primaire

```
CREATE TABLE "ISE_Users" (  
user_id uuid NOT NULL,  
username character varying NOT NULL,  
password character varying NOT NULL  
);
```

```
ALTER TABLE public."ISE_Users" OWNER TO isedb;  
ALTER TABLE ONLY "ISE_Users"  
ADD CONSTRAINT "ISE_Users_pkey" PRIMARY KEY (user_id);
```

2. Exécutez cette requête pour insérer un utilisateur

```
INSERT INTO "ISE_Users" VALUES ('8cc4b9b9-117a-46c4-879e-d764c9685e80', 'user1', 'password1');
```

Ou

```
INSERT INTO "ISE_Users" VALUES (uuid_generate_v1(), 'user1', 'password1');
```

Et apprenez et enregistrez UUID généré d'un nouvel utilisateur avec cette requête

```
SELECT user_id FROM "ISE_Users" WHERE username = 'user1';
```

3. Créez une procédure pour l'authentification de mot de passe de texte brut (utilisée pour la méthode intérieure PAP, EAP-GTC, TACACS)

```
CREATE FUNCTION iseauthuserplainreturnsparameters(ise_username text, ise_password text, OUT  
result integer, OUT ise_group text, OUT acctinfo text, OUT errorstring text) RETURNS record  
LANGUAGE plpgsql IMMUTABLE SECURITY DEFINER  
AS $$  
DECLARE  
c int;  
BEGIN  
select count(*) into c from "ISE_Users" where username = ise_username and password =  
ise_password;  
IF c > 0 THEN  
result := 0;  
ise_group := cast ('11' as text);  
acctinfo := cast ('This is a very good user, give him all access' as text);  
errorstring := cast ('No error' as text);  
else  
result := 3;  
ise_group := cast ('11' as text);  
acctinfo := cast ('User is unknown or invalid password' as text);  
errorstring := cast ('User is unknown or invalid password' as text);  
END IF;  
END;  
$$;
```

```
ALTER FUNCTION public.iseauthuserplainreturnsparameters(ise_username text, ise_password text,  
OUT result integer, OUT ise_group text, OUT acctinfo text, OUT errorstring text) OWNER TO isedb;
```

4. Créez une procédure pour chercher de mot de passe de texte brut (utilisé pour le CHAP, MSCHAPv1/v2, EAP-MD5, LEAP, méthode EAP-MSCHAPv2 intérieure, TACACS)

```
CREATE FUNCTION isefetchpasswordreturnsparameters(ise_username text, OUT result integer, OUT  
ise_group text, OUT acctinfo text, OUT errorstring text, OUT ise_password text) RETURNS record  
LANGUAGE plpgsql IMMUTABLE SECURITY DEFINER  
AS $$  
DECLARE  
c int;
```

```

BEGIN
select count(*) into c from "ISE_Users" where username = ise_username;
IF c > 0 THEN
result := 0;
ise_group := cast ('11' as text);
acctinfo := cast ('This is a very good user, give him all access' as text);
errorstring := cast ('no error' as text);
select password into ise_password from "ISE_Users" where username = ise_username;
else
result := 3;
ise_group := cast ('11' as text);
acctinfo := cast ('User is unknown' as text);
errorstring := cast ('User is unknown' as text);
END IF;
END;
$$;

```

```

ALTER FUNCTION public.isefetchpasswordreturnsparameters(ise_username text, OUT result integer,
OUT ise_group text, OUT acctinfo text, OUT errorstring text, OUT ise_password text) OWNER TO
isedb;

```

5. Créez une procédure pour le nom d'utilisateur de contrôle ou l'ordinateur existe (utilisé pour le MAB, rapide rebranchez du PEAP, de l'EAP-FAST et de l'EAP-TTLS)

```

CREATE FUNCTION iseuserlookupreturnsparameters(ise_username text, OUT result integer, OUT
ise_group text, OUT acctinfo text, OUT errorstring text) RETURNS record
LANGUAGE plpgsql IMMUTABLE SECURITY DEFINER
AS $$
DECLARE
c int;
BEGIN
select count(*) into c from "ISE_Users" where username = ise_username;
IF c > 0 THEN
result := 0;
ise_group := cast ('11' as text);
acctinfo := cast ('good user' as text);
errorstring := cast ('no error' as text);
else
result := 3;
ise_group := cast ('11' as text);
acctinfo := cast ('bad user' as text);
errorstring := cast ('bad password' as text);
END IF;
END;
$$;

```

```

ALTER FUNCTION public.iseuserlookupreturnsparameters(ise_username text, OUT result integer, OUT
ise_group text, OUT acctinfo text, OUT errorstring text) OWNER TO isedb;

```

6. Configurez les procédures sur ISE et les sauvegardez

ODBC Identity Source

General Connection **Stored Procedures** Attributes Groups

Stored procedure type: Returns parameters

Plain text password authentication: iseauthuserplainreturnsparements

Plain text password fetching: isefetchpasswordreturnsparements

Check username or machine exists: iseuserlookupreturnsparements

Fetch groups: []

Fetch attributes: []

Search for MAC Address in format: XX:XX:XX:XX:XX:XX

7. Créez une règle simple d'authentification utilisant ODBC et testez-la

Authentication Policy

Define the Authentication Policy by selecting the protocols that ISE should use to communicate with the network devices, and the identity sources that it should use for authentication. For Policy Export go to [Administration > System > Backup & Restore > Policy Export Page](#)

Policy Type Simple Rule-Based

<input checked="" type="checkbox"/>	MAB	: If Wired_MAB OR
	Wireless_MAB	Allow Protocols : Default Network Access and
<input checked="" type="checkbox"/>	Default	: use Internal Endpoints
<input checked="" type="checkbox"/>	Dot1X	: If Wired_802.1X OR
	Wireless_802.1X	Allow Protocols : Default Network Access and
<input checked="" type="checkbox"/>	Default	: use All_User_ID_Stores
<input checked="" type="checkbox"/>	test_aaa	: If Radius:Service-Type EQUALS Login
	Allow Protocols	: Default Network Access and
<input checked="" type="checkbox"/>	Default	: use pgSQL
<input checked="" type="checkbox"/>	Default Rule (if no match)	: Allow Protocols : Default Network Access and use : All_User_ID_Stores

```
BAHAMUT#test aaa group ISE user1 password1 legacy
Attempting authentication test to server-group ISE using radius
User was successfully authenticated.
```

Overview	
Event	5200 Authentication succeeded
Username	user1
Endpoint Id	
Endpoint Profile	
Authentication Policy	Default >> test_aaa >> Default
Authorization Policy	Default >> Basic_Authenticated_Access
Authorization Result	PermitAccess

Authentication Details	
Source Timestamp	2016-08-26 14:18:28.17
Received Timestamp	2016-08-26 14:18:28.206
Policy Server	vltunov-ise21
Event	5200 Authentication succeeded
Username	user1
Authentication Identity Store	pgSQL
Authentication Method	PAP_ASCII
Authentication Protocol	PAP_ASCII

Steps

```

11001 Received RADIUS Access-Request
11017 RADIUS created a new session
11117 Generated a new session ID for a 3rd party NAD
15049 Evaluating Policy Group
15008 Evaluating Service Selection Policy
15048 Queried PIP - Normalised Radius RadiusFlowType (2 times)
15048 Queried PIP - Radius Service-Type
15048 Queried PIP - Normalised Radius RadiusFlowType (2 times)
15004 Matched rule - test_aaa
15041 Evaluating Identity Policy
15006 Matched Default Rule
15013 Selected Identity Source - pgSQL
24852 Perform plain text password authentication in external ODBC database - pgSQL
24849 Connecting to external ODBC database - pgSQL
24850 Successfully connected to external ODBC database - pgSQL
24850 Expect external ODBC database stored procedure to return results in output parameters - pgSQL
22037 Authentication Passed
15036 Evaluating Authorization Policy
15048 Queried PIP - Normalised Radius RadiusFlowType (4 times)
15048 Queried PIP - EndPoints.LogicalProfile
15048 Queried PIP - Network Access.AuthenticationStatus
15004 Matched rule - Basic_Authenticated_Access
15016 Selected Authorization Profile - PermitAccess
11002 Returned RADIUS Access-Accept

```

Étape 4. Configurez la récupération de groupe

1. Créez les tables contenant des groupes d'utilisateurs et des autres utilisées pour le mappage multiple

```

CREATE TABLE "Groups" (
group_id uuid NOT NULL,
group_name character varying(255) NOT NULL,
group_description text
);

```

```

ALTER TABLE public."Groups" OWNER TO isedb;

```

```

ALTER TABLE ONLY "Groups"
ADD CONSTRAINT "Groups_pkey" PRIMARY KEY (group_id);

```

```

CREATE TABLE "User_Groups_Mapping" (
user_id uuid,
group_id uuid
);

```

```

ALTER TABLE public."User_Groups_Mapping" OWNER TO isedb;

```

```

ALTER TABLE ONLY "User_Groups_Mapping"
ADD CONSTRAINT "User_Groups_Mapping_group_id_fkey" FOREIGN KEY (group_id) REFERENCES
"Groups"(group_id) ON UPDATE CASCADE ON DELETE CASCADE;

```

```

ALTER TABLE ONLY "User_Groups_Mapping"
ADD CONSTRAINT "User_Groups_Mapping_user_id_fkey" FOREIGN KEY (user_id) REFERENCES
"ISE_Users"(user_id) ON UPDATE CASCADE ON DELETE CASCADE;

```

2. Ajoutez les groupes et les mappages, de sorte qu'user1 appartienne à deux groupes

```

INSERT INTO "Groups" VALUES ('f7dfec5c-bd06-4703-9de0-4d334ea5ec02', 'Admins', 'Group for administrators');
INSERT INTO "Groups" VALUES ('51fc0ccd-caf8-4585-ba20-6596948c879d', 'Users', 'Group for users');

```

```
INSERT INTO "Groups" VALUES ('7b7e72bc-ea22-470c-8578-1dd86b1a1843', 'Laptops', 'Group for users with laptops');
```

```
INSERT INTO "User_Groups_Mapping" VALUES ('8cc4b9b9-117a-46c4-879e-d764c9685e80', 'f7dfec5c-bd06-4703-9de0-4d334ea5ec02');
```

```
INSERT INTO "User_Groups_Mapping" VALUES ('8cc4b9b9-117a-46c4-879e-d764c9685e80', '7b7e72bc-ea22-470c-8578-1dd86b1a1843');
```

Ou générez nouvel UUIDs, toutefois vous devrez les apprendre avec des requêtes **CHOISIES**.

3. Créez le type de retour et une procédure de récupération de groupe

```
CREATE TYPE g4type AS (  
result integer,  
group_n text  
);
```

```
ALTER TYPE public.g4type OWNER TO isedb;
```

```
CREATE FUNCTION isegroupsh(ise_username text) RETURNS SETOF g4type  
LANGUAGE plpgsql IMMUTABLE SECURITY DEFINER  
AS $$  
DECLARE  
c int;  
i int;  
r g4type%rowtype;  
BEGIN  
if ise_username = '*' then  
for r in select 0, cast(group_name as text) from "Groups"  
loop  
return next r;  
end loop;  
else  
select count(*) into c from "ISE_Users" where username = ise_username;  
IF c > 0 THEN  
for r in select 0, cast(group_name as text) from "Groups" where group_id in (  
select group_ID from "User_Groups_Mapping" where "User_Groups_Mapping".user_id IN (  
select user_id from "ISE_Users" where username = ise_username  
) )  
loop  
return next r;  
end loop;  
else  
return query select 1, cast ('' as text);  
END IF;  
end if;  
END;  
$$;
```

```
ALTER FUNCTION public.isegroupsh(ise_username text) OWNER TO isedb;
```

4. Tracez-le pour chercher des groupes

ODBC Identity Source

General Connection **Stored Procedures** Attributes Groups

Stored procedure type: Returns parameters

Plain text password authentication: iseauthuserplainreturnsparemeters

Plain text password fetching: isefetchpasswordreturnsparemeters

Check username or machine exists: iseuserlookupreturnsparemeters

Fetch groups: isegroupsh

Fetch attributes: iseattrsh

Search for MAC Address in format: XX:XX:XX:XX:XX:XX

5. Cherchez les groupes et ajoutez-les dans la **source d'identité ODBC**

ODBC Identity Source

General Connection Stored Procedures Attributes **Groups**

Edit + Add X Delete

Name	Name in ISE
No data available	

Select Groups from ODBC

Sample User or Machine: * Retrieve Groups

<input checked="" type="checkbox"/>	Name	Name in ISE
<input checked="" type="checkbox"/>	Admins	Admins
<input checked="" type="checkbox"/>	Users	Users
<input checked="" type="checkbox"/>	Laptops	Laptops

OK Cancel

6. Ajoutez un autre utilisateur qui n'appartient pas à tout groupe

```
INSERT INTO "ISE_Users" VALUES ('592136bb-9c47-49ff-8eca-9adfb2016b1c', 'user2', 'password2');
```

7. Créez une **stratégie d'autorisation de test** et testez-la

<input checked="" type="checkbox"/>	ODBC check Group	if	pgSQL-ExternalGroups EQUALS Admins	then	PermitAccess
<input checked="" type="checkbox"/>	Default	if no matches, then			DenyAccess

BAHAMUT#test aaa group ISE user1 password1 legacy
 Attempting authentication test to server-group ISE using radius
 User was successfully authenticated.

BAHAMUT#test aaa group ISE user2 password2 legacy
 Attempting authentication test to server-group ISE using radius
 User authentication request was rejected by server.

SelectedAuthenticationIdentityStores	pgSQL
AuthorizationPolicyMatchedRule	ODBC check Group
CPMSessionID	0a301a321uM9iabemtWC3JmOxM0PEPNRCy44aEudtrNg2ajmJGg
ISEPolicySetName	Default
AllowedProtocolMatchedRule	test_aaa
IdentitySelectionMatchedRule	Default
Network Device Profile	Cisco
Location	Location#All Locations
Device Type	Device Type#All Device Types
ExternalGroups	Admins
ExternalGroups	Laptops
RADIUS Username	user1

Étape 5. Configurez la récupération d'attributs

1. Afin de simplifier cet exemple, une table plate est utilisée pour des attributs

```
CREATE TABLE "User_Attributes" (
  user_id uuid,
  attribute_name character varying(255),
  attribute_value character varying(255)
);
```

```
ALTER TABLE public."User_Attributes" OWNER TO isedb;
```

```
ALTER TABLE ONLY "User_Attributes"
ADD CONSTRAINT "User_Attributes_user_id_fkey" FOREIGN KEY (user_id) REFERENCES
"ISE_Users"(user_id) ON UPDATE CASCADE ON DELETE CASCADE;
```

2. Créez un attribut pour chacun des deux utilisateurs

```
INSERT INTO "User_Attributes" VALUES ('8cc4b9b9-117a-46c4-879e-d764c9685e80', 'SecurityLevel',
'10');
```

```
INSERT INTO "User_Attributes" VALUES ('592136bb-9c47-49ff-8eca-9adfb2016b1c', 'SecurityLevel', '5');
```

```
INSERT INTO "User_Attributes" VALUES ('592136bb-9c47-49ff-8eca-9adfb2016b1c', 'IdleTimeout', '5');
```

3. Créez un type de retour et une procédure stockée

```
CREATE TYPE a4type AS (  
result integer,  
attr_name text,  
attr_value text  
);
```

```
ALTER TYPE public.a4type OWNER TO isedb;
```

```
CREATE FUNCTION iseattrsh(ise_username text) RETURNS SETOF a4type  
LANGUAGE plpgsql IMMUTABLE SECURITY DEFINER  
AS $$  
DECLARE  
c int;  
r a4type%rowtype;  
BEGIN  
select count(*) into c from "ISE_Users" where username = ise_username;  
IF c > 0 THEN  
for r in select 0, cast(s.attribute_name as text), cast(s.attribute_value as text) from  
"User_Attributes" as s where user_id in(SELECT user_id from "ISE_Users" where username =  
ise_username)  
loop  
return next r;  
end loop;  
else  
return query select 1, cast ('' as text);  
END IF;  
END;  
$$;
```

```
ALTER FUNCTION public.iseattrsh(ise_username text) OWNER TO isedb;
```

4. Tracez-le pour chercher des attributs

[ODBC List > pgSQL](#)

ODBC Identity Source

General

Connection

Stored Procedures

Attributes

Groups

Stored procedure type

Plain text password authentication ⓘ ⊕

Plain text password fetching ⓘ ⊕

Check username or machine exists ⓘ ⊕

Fetch groups ⓘ ⊕

Fetch attributes ⓘ ⊕

Search for MAC Address in format ⓘ

5. Cherchez les attributs

ODBC List > pgSQL

ODBC Identity Source

General Connection Stored Procedures **Attributes** Groups

Edit + Add - Delete

Name	Type	Default Value	Name in ISE
No data available			

Select Attributes from ODBC

Sample User or Machine:

<input checked="" type="checkbox"/>	Name	Type	Default Value	Name in ISE
<input checked="" type="checkbox"/>	SecurityLevel	STRING	5	SecurityLevel
<input checked="" type="checkbox"/>	IdleTimeout	STRING	5	IdleTimeout

6. Ajustez les stratégies ISE et testez-les

<input checked="" type="checkbox"/>	ODBC all access	if (pgSQL.ExternalGroups EQUALS Admins AND pgSQL.SecurityLevel EQUALS 10)	then PermitAccess
<input checked="" type="checkbox"/>	ODBC security 5	if pgSQL.SecurityLevel EQUALS 5	then Sec-5

Status	Details	Repeat ...	Identity	End...	Endp...	Authenticati...	Authorization Policy	Authorizati...	IP
<input checked="" type="checkbox"/>			<input type="text" value="Identity"/>	<input type="text" value="Endpc"/>	<input type="text" value="Endpoi"/>	<input type="text" value="Authentication"/>	<input type="text" value="Authorization Policy"/>	<input type="text" value="Authorization F"/>	<input type="text" value="IP"/>
<input checked="" type="checkbox"/>			user2			Default >> te...	Default >> ODBC security 5	Sec-5	
<input checked="" type="checkbox"/>			user1			Default >> te...	Default >> ODBC all access	PermitAccess	

Vérifiez

Vous devriez maintenant pouvoir authentifier des utilisateurs contre ODBC et récupérer leurs groupes et attributs.

Exemple :

Overview	
Event	5200 Authentication succeeded
Username	user1
Endpoint ID	
Endpoint Profile	
Authentication Policy	Default == Int_Lan == Default
Authorization Policy	Default == ODBC all access
Authoritative Result	PermAccess

Authentication Details	
Source Timestamp	2016-08-28 13:37:43.957
Received Timestamp	2016-08-28 13:37:43.958
Policy Server	vlmwr-0a21
Event	5200 Authentication succeeded
Username	user1
Authentication Identity Store	pgSQL
Authentication Method	PAP_PDCB
Authentication Protocol	PAP_PDCB
Service Type	Login
Network Device	Infanet
Device Type	All Device Types
Location	All Locations
NAS IPv4 Address	10.48.44.114
NAS Port Type	Async
Authoritative Profile	PermAccess
Response Time	148

Other Attributes	
ConfigVersion	103
DestinationPort	1812
Protocol	Radius
NetworkDeviceProfileName	Cisco
NetworkDeviceProfileID	403ea8b0-7a27-41c3-b036-27964031a09d
IsThirdPartyDevice? low	False
Acta SessionID	vlmwr-0a210570121913812
SelectedAuthenticationIdentityStores	pgSQL
AuthorizationPolicyMatchedRule	ODBC all access
CPM SessionID	9a301a23f0-g048GwrgLFF2fCvV04e1wqPKQu0EM0g
VSE PolicySetName	Default
AllowedProtocolMatchedRule	Int_Lan
IdentitySelectorMatchedRule	Default
Network Device Profile	Cisco
Location	Location# All Locations
Device Type	Device Type# All Device Types
ExternalGroups	Admins
ExternalGroups	Laptops
SecurityLevel	10
RADIUS Username	user1

Dépannez

Si la connexion n'est pas réussie sur la queue de `prft-management.log` d'application de `show logging` de commande d'utilisation ISE tout en tentant de se connecter.

Exemple des qualifications fausses :

```
2016-08-28 13:55:47,017 WARN [admin-http-pool1372][] cisco.cpm.odbcidstore.impl.PostgresDbAccess
-:admin::- Connection to ODBC DB failed. Exception: org.postgresql.util.PSQLException: FATAL:
password authentication failed for u
```

```

ser "isedb_wrong"
org.postgresql.util.PSQLException: FATAL: password authentication failed for user "isedb_wrong"
at org.postgresql.Driver$ConnectThread.getResult(Driver.java:365)
at org.postgresql.Driver.connect(Driver.java:288)
at java.sql.DriverManager.getConnection(DriverManager.java:664)
at java.sql.DriverManager.getConnection(DriverManager.java:208)
at com.cisco.cpm.odbcidstore.impl.PostgresDbAccess.connect(PostgresDbAccess.java:46)
at com.cisco.cpm.odbcidstore.impl.OdbcConnection.connect(OdbcConnection.java:72)
at com.cisco.cpm.odbcidstore.impl.OdbcIdStore.performTest(OdbcIdStore.java:377)
at
com.cisco.cpm.odbcidstore.impl.OdbcIdStore.testConnectionAndConfiguration(OdbcIdStore.java:469)
at
com.cisco.cpm.odbcidstore.impl.OdbcIdStoreManager.testConnectionAndConfiguration(OdbcIdStoreManager.java:84)
at com.cisco.cpm.admin.ac.actions.ODBCLPInputAction.testConnection(ODBCLPInputAction.java:749)

```

Exemple du nom faux de DB :

```

2016-08-28 13:53:43,174 WARN [admin-http-pool1372][] cisco.cpm.odbcidstore.impl.PostgresDbAccess
-:admin:- Connection to ODBC DB failed. Exception: org.postgresql.util.PSQLException: FATAL:
database "isedb_wrong" does not exist
t
org.postgresql.util.PSQLException: FATAL: database "isedb_wrong" does not exist
at org.postgresql.Driver$ConnectThread.getResult(Driver.java:365)
at org.postgresql.Driver.connect(Driver.java:288)
at java.sql.DriverManager.getConnection(DriverManager.java:664)
at java.sql.DriverManager.getConnection(DriverManager.java:208)
at com.cisco.cpm.odbcidstore.impl.PostgresDbAccess.connect(PostgresDbAccess.java:46)
at com.cisco.cpm.odbcidstore.impl.OdbcConnection.connect(OdbcConnection.java:72)
at com.cisco.cpm.odbcidstore.impl.OdbcIdStore.performTest(OdbcIdStore.java:377)
at
com.cisco.cpm.odbcidstore.impl.OdbcIdStore.testConnectionAndConfiguration(OdbcIdStore.java:469)
at
com.cisco.cpm.odbcidstore.impl.OdbcIdStoreManager.testConnectionAndConfiguration(OdbcIdStoreManager.java:84)
at com.cisco.cpm.admin.ac.actions.ODBCLPInputAction.testConnection(ODBCLPInputAction.java:749)

```

Afin de dépanner des exécutions de DB, enable se connectant l'odbc-id-mémoire de composants **POUR DÉBUGGER** de niveau sous la **gestion > le système > se connectant > configuration de log de debug**.

Des logs sont placés dans le fichier de **prrt-management.log**.

Exemple pour user1 :

```

2016-08-28 14:01:01,116 DEBUG [Thread-26349][] cisco.cpm.odbcidstore.impl.OdbcIdStore -:::- ODBC
ID Store Operation: Authenticate Plain Text Password. Username=user1,
SessionID=0a301a320uqzqoKTrY02KoCjdWN6PLZtBX1/vhDXxN9nQTBFM8g
2016-08-28 14:01:01,118 DEBUG [Thread-26349][] cisco.cpm.odbcidstore.impl.CustomerLog -:::-
Write customer log message: 24852
2016-08-28 14:01:01,119 DEBUG [Thread-26349][] cisco.cpm.odbcidstore.impl.OdbcConnectionPool -
:::- OdbcConnectionPool - get connection
2016-08-28 14:01:01,119 DEBUG [Thread-26349][] cisco.cpm.odbcidstore.impl.OdbcConnectionPool -
:::- OdbcConnectionPool - use existing connection
2016-08-28 14:01:01,119 DEBUG [Thread-26349][] cisco.cpm.odbcidstore.impl.OdbcConnectionPool -
:::- OdbcConnectionPool - connections in use: 1
2016-08-28 14:01:01,119 DEBUG [Thread-26349][] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Authenticate plain text password
2016-08-28 14:01:01,119 DEBUG [Thread-26349][] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Prepare stored procedure call, procname=iseauthuserplainreturnparameters
2016-08-28 14:01:01,119 DEBUG [Thread-26349][] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-

```

Using output parameters to obtain stored procedure result values
2016-08-28 14:01:01,119 DEBUG [Thread-26349][[] cisco.cpm.odbcidstore.impl.CustomerLog -:::-
Write customer log message: 24856
2016-08-28 14:01:01,119 DEBUG [Thread-26349][[] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Text: {call iseauthuserplainreturnsparameters(?, ?, ?, ?, ?, ?)}
2016-08-28 14:01:01,119 DEBUG [Thread-26349][[] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Setup stored procedure input parameters, username=user1, password=***
2016-08-28 14:01:01,119 DEBUG [Thread-26349][[] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Setup stored procedure output parameters
2016-08-28 14:01:01,119 DEBUG [Thread-26349][[] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Execute stored procedure call
2016-08-28 14:01:01,121 DEBUG [Thread-26349][[] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Process stored procedure results
2016-08-28 14:01:01,121 DEBUG [Thread-26349][[] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Obtain stored procedure results from output parameters
2016-08-28 14:01:01,121 DEBUG [Thread-26349][[] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Results successfully parsed from output parameters
2016-08-28 14:01:01,121 DEBUG [Thread-26349][[] cisco.cpm.odbcidstore.impl.OdbcConnectionPool -
:::- OdbcConnectionPool - release connection
2016-08-28 14:01:01,121 DEBUG [Thread-26349][[] cisco.cpm.odbcidstore.impl.OdbcConnectionPool -
:::- OdbcConnectionPool - connections in use: 0
2016-08-28 14:01:01,121 DEBUG [Thread-26349][[] cisco.cpm.odbcidstore.impl.OdbcIdStore -:::- Call
to ODBC DB succeeded
2016-08-28 14:01:01,121 DEBUG [Thread-26349][[] cisco.cpm.odbcidstore.impl.OdbcAuthResult -:::-
Authentication result: code=0, Connection succeeded=false, odbcDbErrorString=No error,
odbcStoredProcedureCustomerErrorString=null, ac
countInfo=This is a very good user, give him all access, group=11
2016-08-28 14:01:01,121 DEBUG [Thread-26349][[] cisco.cpm.odbcidstore.impl.CustomerLog -:::-
Write customer log message: 24853
2016-08-28 14:01:01,129 DEBUG [Thread-3076][[] cisco.cpm.odbcidstore.impl.OdbcIdStore -:::- ODBC
ID Store Operation: **Get all user groups**. Username=user1,
SessionID=0a301a320uqzqokTrY02KoCjdWN6PlZtBX1/vhDXxN9nQTBFM8g
2016-08-28 14:01:01,131 DEBUG [Thread-3076][[] cisco.cpm.odbcidstore.impl.OdbcIdStore -:::- ODBC
ID Store Operation: **Fetch user groups**. Username=user1,
SessionID=0a301a320uqzqokTrY02KoCjdWN6PlZtBX1/vhDXxN9nQTBFM8g
2016-08-28 14:01:01,131 DEBUG [Thread-3076][[] cisco.cpm.odbcidstore.impl.CustomerLog -:::- Write
customer log message: 24869
2016-08-28 14:01:01,132 DEBUG [Thread-3076][[] cisco.cpm.odbcidstore.impl.OdbcConnectionPool -
:::- OdbcConnectionPool - get connection
2016-08-28 14:01:01,132 DEBUG [Thread-3076][[] cisco.cpm.odbcidstore.impl.OdbcConnectionPool -
:::- OdbcConnectionPool - use existing connection
2016-08-28 14:01:01,132 DEBUG [Thread-3076][[] cisco.cpm.odbcidstore.impl.OdbcConnectionPool -
:::- OdbcConnectionPool - connections in use: 1
2016-08-28 14:01:01,132 DEBUG [Thread-3076][[] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Fetch user groups
2016-08-28 14:01:01,132 DEBUG [Thread-3076][[] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Prepare stored procedure call, procname=**isegroupsh**
2016-08-28 14:01:01,132 DEBUG [Thread-3076][[] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Text: {call isegroupsh(?)}
2016-08-28 14:01:01,132 DEBUG [Thread-3076][[] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Setup stored procedure input parameters, username=user1
2016-08-28 14:01:01,132 DEBUG [Thread-3076][[] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Execute stored procedure call
2016-08-28 14:01:01,134 DEBUG [Thread-3076][[] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Process stored procedure results
2016-08-28 14:01:01,135 DEBUG [Thread-3076][[] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Received result recordset, total number of columns=2
2016-08-28 14:01:01,135 DEBUG [Thread-3076][[] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
POSTGRES case, first column holds the result param value
2016-08-28 14:01:01,135 DEBUG [Thread-3076][[] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
According to column number expect multiple rows (vertical attributes/groups returned result)
2016-08-28 14:01:01,135 DEBUG [Thread-3076][[] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Fetched data: **ExternalGroup=Admins**
2016-08-28 14:01:01,135 DEBUG [Thread-3076][[] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-

Fetch data: **ExternalGroup=Laptops**

```
2016-08-28 14:01:01,135 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Results successfully parsed from recordset
2016-08-28 14:01:01,135 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Result code indicates success
2016-08-28 14:01:01,135 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcConnectionPool -
:::- OdbcConnectionPool - release connection
2016-08-28 14:01:01,135 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcConnectionPool -
:::- OdbcConnectionPool - connections in use: 0
2016-08-28 14:01:01,135 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcIdStore -:::- Call
to ODBC DB succeeded
2016-08-28 14:01:01,135 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.CustomerLog -:::- Write
customer log message: 24870
2016-08-28 14:01:01,135 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcIdStore -:::- ODBC
ID Store Operation: Get all user groups. Got groups...
2016-08-28 14:01:01,135 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcIdStore -:::- ODBC
ID Store Operation: Get all user groups. Got groups(0) = Admins
2016-08-28 14:01:01,135 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcIdStore -:::- ODBC
ID Store Operation: Get all user groups. Setting Internal groups(0) = Admins
2016-08-28 14:01:01,135 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcIdStore -:::- ODBC
ID Store Operation: Get all user groups. Got groups(1) = Laptops
2016-08-28 14:01:01,135 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcIdStore -:::- ODBC
ID Store Operation: Get all user groups. Setting Internal groups(1) = Laptops
2016-08-28 14:01:01,135 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcIdStore -:::- ODBC
ID Store Operation: Get all user groups. Username=user1, ExternalGroups=[Admins, Laptops]
2016-08-28 14:01:01,135 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcIdStore -:::- ODBC
ID Store Operation: Fetch user attributes. Username=user1,
SessionID=0a301a320uqzqoKTrY02KoCjdWN6PlZtBX1/vhDXxN9nQTBFM8g
2016-08-28 14:01:01,135 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.CustomerLog -:::- Write
customer log message: 24872
2016-08-28 14:01:01,135 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcConnectionPool -
:::- OdbcConnectionPool - get connection
2016-08-28 14:01:01,135 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcConnectionPool -
:::- OdbcConnectionPool - use existing connection
2016-08-28 14:01:01,135 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcConnectionPool -
:::- OdbcConnectionPool - connections in use: 1
2016-08-28 14:01:01,135 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Fetch user attributes
2016-08-28 14:01:01,135 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Prepare stored procedure call, procname=iseattrsh
2016-08-28 14:01:01,135 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Text: {call iseattrsh(?)}
```

```
2016-08-28 14:01:01,135 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Setup stored procedure input parameters, username=user1
2016-08-28 14:01:01,135 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Execute stored procedure call
2016-08-28 14:01:01,140 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Process stored procedure results
2016-08-28 14:01:01,140 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Received result recordset, total number of columns=3
2016-08-28 14:01:01,140 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
POSTGRES case, first column holds the result param value
2016-08-28 14:01:01,140 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
According to column number expect multiple rows (vertical attributes/groups returned result)
2016-08-28 14:01:01,140 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Fetch data: SecurityLevel=10
2016-08-28 14:01:01,140 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Results successfully parsed from recordset
2016-08-28 14:01:01,140 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcConnection -:::-
Result code indicates success
2016-08-28 14:01:01,140 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcConnectionPool -
:::- OdbcConnectionPool - release connection
2016-08-28 14:01:01,140 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcConnectionPool -
:::- OdbcConnectionPool - connections in use: 0
```



```
2016-08-28 14:01:01,140 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcIdStore -:::- Call
to ODBC DB succeeded
2016-08-28 14:01:01,140 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.CustomerLog -:::- Write
customer log message: 24873
2016-08-28 14:01:01,141 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcIdStore -:::- ODBC
ID Store Operation: Get all user attrs. Username=user1, Setting pgSQL.SecurityLevel to 10
2016-08-28 14:01:01,141 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcIdStore -:::- ODBC
ID Store Operation: Get all user attrs. Username=user1, Setting IdleTimeout to default value : 5
2016-08-28 14:01:01,141 DEBUG [Thread-3076][] cisco.cpm.odbcidstore.impl.OdbcIdStore -:::- ODBC
ID Store Operation: Get all user attrs. Username=user1, Setting pgSQL.IdleTimeout to 5
```

Références

- [Guide d'administration du Logiciel Cisco Identity Services Engine 2.1 - Configuration ODBC](#)
- [Configurez ISE 2.1 avec le MS SQL utilisant ODBC](#)
- [PostgreSQL : Documentation](#)