Finding the source of failed login attempts. (Doc ID 352389.1)

In this Document

APPLIES TO:

Oracle Database - Enterprise Edition - Version 9.0.1.4 to 11.2.0.4 [Release 9.0.1 to 11.2]
Information in this document applies to any platform.
Oracle Server Enterprise Edition - Version: 9.0.1.4 to 11.2.0.4

PURPOSE

This troubleshooting guide is aimed at providing information on how to trace suspect logins that have failed with ora-1017, logon denied during database authentication. Note that a malicious origin is in most cases not the cause of an unsuccessful login. It can be simply a typo in the username or password, a cron job with an invalid password in it or even an Oracle client program that was badly configured (for example see referenced note 267401.1). Anyway these unsuccessful connect attempts can be a nuisance, especially when a password management policy is in place that has a limit on failed_login_attempts and cause the ora-28000 "the account is locked" error.

TROUBLESHOOTING STEPS

Check DBA_USERS

If the ACCOUNT_STATUS for the user is LOCKED, this means the account was locked by a DBA doing *ALTER USER &username ACCOUNT LOCK;* if the ACCOUNT_STATUS is LOCKED(TIMED) this means it was locked because of exceeding the number of allowed FAILED_LOGIN_ATTEMPTS, this is true even if the PASSWORD_LOCK_TIME is set to unlimited. An account will never get locked due to password expiration. Before the account is actually locked you can inspect the number of failed login attempts so far by checking the USER$.LCOUNT column:

```
select name,lcount from user$ where name='&USERNAME';
```

Audit unsuccessful logins

At first establish standard auditing by setting audit_trail = db and issue:

AUDIT SESSION WHENEVER NOT SUCCESSFUL;

The audit records for unsuccessful logon attempts from 'last week' can be found as follows:

```
select username,
os_username,
userhost,
client_id,
trunc(timestamp),
count(*) failed_logins
from dba_audit_trail
where returncode=1017 and --1017 is invalid username/password
timestamp > sysdate -7
group by username,os_username,userhost, client_id,trunc(timestamp);
```

Comment: The USERHOST column is only populated with the *Client Host machine name* as of 10G, in earlier versions this was the *Numeric instance ID for the Oracle instance from which the user is accessing the database* in a RAC environment.

Optionally select more columns, these may however not provide you with relevant information, your reason for checking this note may be that you found some entries here and need more information. Try to correlate the DBA_USERS.LOCK_DATE with the time scripts or batch jobs are being run.

Set an event in the "init.ora" parameter file

Set the following event in your parameter file to dump a trace file whenever an ORA-1017 is generated:

```
event = "1017 trace name errorstack level 10"
```

Alternatively you can issue the following command as a privileged user, this will only affect new processes so may not work in a Shared Server environment:

```
alter system set events '1017 trace name errorstack level 10';
```

This will produce a trace file in user_dump_dest whenever someone attempts an invalid username / password , since the trace is requested at level 10, it will include a section labeled PROCESS STATE that includes trace information like:

```
O/S info: user: userx, term: pts/1, ospid: ***** , machine: *****
program: sqlplus@***** (TNS V1-V3)
application name: sqlplus@**** (TNS V1-V3), hash value=0
last wait for 'SQL*Net message from client' blocking sess=0x0 seq=2 wait_tim
e=5570 seconds since wait started=0
```

In this case it was an sqlplus client started by OS user 'userx' that started the client session. The section Call Stack Trace may aid support in further diagnosing the issue. Tip: If the OS user or program is 'oracle' the connection may originate from a Database Link.

Use a trigger to capture additional information

The following trigger code can be used to gather additional information about unsuccessful login attempts, it is recommended to integrate this code into an existing trigger if you already have a trigger for this triggering event instead of having more triggers on the same event, note this is just an example providing some essential information, it can be changed or modified at will, for example you may prefer to log the entries in a table instead of the alert.log file, it appears this trigger fires even if the session is not authenticated, please keep the trigger code as simply as possible as to minimize the performance impact.

```
-- sample trigger to write diagnostic info to alert.log
-- for failed login attempts (ora-1017)
```

```
create or replace trigger logon_denied_to_alert
after servererror on database
declare
message varchar2(256);
IP varchar2(15);
v_os_user varchar2(80);
v_module varchar2(50);
v_action varchar2(50);
v_pid varchar2(10);
v_sid number;
v_program varchar2(48);
v_client_id VARCHAR2(64);
begin
IF (ora_is_servererror(1017)) THEN

-- get IP for remote connections:
if sys_context('userenv','network_protocol') = 'TCP' then
IP := sys_context('userenv','ip_address');
end if;

select distinct sid into v_sid from sys.v_$mystat;
SELECT p.SPID, v.PROGRAM into v_pid, v_program
FROM V$PROCESS p, V$SESSION v
WHERE p.ADDR = v.PADDR AND v.sid = v_sid;

v_os_user := sys_context('userenv','os_user');
dbms_application_info.READ_MODULE(v_module,v_action);

v_client_id := sys_context('userenv','client_identifier');

message:= to_char(sysdate,'Dy Mon dd HH24:MI:SS YYYY')||
' logon denied '|| 'IP ='||nvl(IP,'localhost')||' pid = '||v_pid||
' os user = '||v_os_user||' client id = '||v_client_id||
' with program= '||v_program||' module ='||v_module||' action='||v_action;

sys.dbms_system.ksdwrt(2,message);

-- remove comments from next line to let it hang for 5 minutes
-- to be able to do more diagnostics on the operating system:
-- sys.dbms_lock.sleep(300);
end if;
end;
/
-- end trigger
```

Some sample output from the alert.log looks like:

```
Fri May 20 10:00:50 2011 Fri May 20 10:00:50 2011 logon denied IP = localhost pid = **** os user = oracle client ic
```

More attributes may be found in note 120797.1 , however please consider when this trigger fires on an unsuccessful logon attempt, not all session specific information may be available as it would be for an authenticated session.

Use SQLNET Tracing to gather detailed information

An sqlnet trace can provide you with even more details about the connection attempt, use this only if the above does not provide you with enough information, since it will be hard to find what you are looking for if you enable sqlnet tracing, to enable it create or edit the server side sqlnet.ora file and put in the following parameters:

```
# server side sqlnet trace parameters

trace_level_server = 16

trace_file_server=server

trace_directory_server = <any directory on a volume with enough freespace>
```

As a last resort you may want to set your hopes on a packet sniffer on the network or operating system level, the use of such tools however is beyond the scope of this article, you may want to consult your local network administrator about this possibility.

Note: If you are seeing your RDBMS account getting locked out(with an ORA-28000 "The account is locked." error) even though user never provided the wrong password
FAILED_LOGIN_ATTEMPTS times consecutively, and there were no ORA-1017 errors in the audit trail for user account, then a known bug can be cause for this and will have to apply patch to fix the issue Bug 30210753.

Please apply patch and verify if the issue gets fixed.

REFERENCES

NOTE:1309738.1 - High 'library cache lock' Wait Time Due to Invalid Login Attempts
NOTE:114930.1 - Oracle Password Management Policy
NOTE:120797.1 - How to Determine Client IP-address,Language & Territory and Username for Current Session
NOTE:221944.1 - How to Audit Potential Attempts to Break a Username/Password
NOTE:259387.1 - How to Change DBSNMP Password in Database 10g and 11g Monitored by DB Control
NOTE:260111.1 - How to Interpret the ACCOUNT_STATUS Column in DBA_USERS
NOTE:284344.1 - DBA_USERS.ACCOUNT_STATUS shows LOCKED after FAILED_LOGIN_ATTEMPTS Is Breached