How to analyze the log files to identify single-sign on (SSO) issues

- Parallels Secure Workspace

Resolution

First, validate whether **pre-authentication** works. This is the phase where users are first redirected to an external Identity Provider (IdP), where they sign on. A common example is a SAML integration, where end users navigating to the Workspace are redirected to Microsoft Azure where they sign in using their Microsoft account. After that, they're redirected to the Workspace and are asked to enter their password.

If the above is working but **single sign-on (SSO)** is still not working, it's best to analyze the log files (see Download logs from an appliance).

SSO has a lot of prerequisites to set up, so it's easy to forget a step. Rather than going over a long checklist, the log files usually narrow down the issue.
Mind that after addressing the (potential) cause, it's best to re-generate and re-check the log files; as perhaps the issue is unresolved or another one occurs.

Only when experiencing single sign-on (SSO) issues, it will be possible to see something similar to the snippet below in the **awingu-worker-smc.service.log.**

It is a very verbose output around the time (in UTC format) of the sign-in attempt.

It's best to read this in reverse order, from bottom to top.

```
2021-09-10 11:41:32.493618 awingu awingu-worker-smc.service[python3:1194]:
Generating a 2048 bit RSA private key
2021-09-10 11:41:32.510919 awingu awingu-worker-smc.service[python3:1194]:
..............+++
2021-09-10 11:41:32.512982 awingu awingu-worker-smc.service[python3:1194]:
.+++
2021-09-10 11:41:32.513234 awingu awingu-worker-smc.service[python3:1194]:
unable to write 'random state'
2021-09-10 11:41:32.513385 awingu awingu-worker-smc.service[python3:1194]:
writing new private key to 'private_key.pem'
2021-09-10 11:41:32.513532 awingu awingu-worker-smc.service[python3:1194]:
-----
2021-09-10 11:41:32.561966 awingu awingu-worker-smc.service[python3:1194]:
writing RSA key
2021-09-10 11:41:32.685396 awingu awingu-worker-smc.service[manage.py:16603]:
Password for someuser\@somedomain.org@SOMEDOMAIN.ORG:

2021-09-10 11:41:32.685611 awingu awingu-worker-smc.service[manage.py:16603]:
Using specified cache:
/etc/awingu/domains/SOMEAWINGUDOMAIN/0f5854e8-8775-4604-a4fd-919391b078f4/kerberos/k
Using principal: someuser\@somedomain.org@SOMEDOMAIN.ORG
PA Option X509_user_identity =
FILE:/etc/awingu/domains/SOMEAWINGUDOMAIN/0f5854e8-8775-4604-a4fd-919391b078f4/certi
[24324] 1631274092.590565: Getting initial credentials for
```

```
someuser\@somedomain.org@SOMEDOMAIN.ORG
[24324] 1631274092.590567: Sending unauthenticated request
[24324] 1631274092.590568: Sending request (211 bytes) to SOMEDOMAIN.ORG
[24324] 1631274092.590569: Resolving hostname 10.1.2.3
[24324] 1631274092.590570: Sending initial UDP request to dgram 10.1.2.3:88
[24324] 1631274092.590571: Received answer (202 bytes) from dgram 10.1.2.3:88
[24324] 1631274092.590572: Sending DNS URI query for
_kerberos.SOMEDOMAIN.ORG.
[24324] 1631274092.590573: No URI records found
[24324] 1631274092.590574: Sending DNS SRV query for
_kerberos-master._udp.SOMEDOMAIN.ORG.
[24324] 1631274092.590575: Sending DNS SRV query for
_kerberos-master._tcp.SOMEDOMAIN.ORG.
[24324] 1631274092.590576: No SRV records found
[24324] 1631274092.590577: Response was not from master KDC
[24324] 1631274092.590578: Received error from KDC: -1765328359/Additional
pre-authentication required
[24324] 1631274092.590581: Preauthenticating using KDC method data
[24324] 1631274092.590582: Processing preauth types: PA-PK-AS-REQ (16),
PA-PK-AS-REP_OLD (15), PA-ETYPE-INFO2 (19), PA-ENC-TIMESTAMP (2)
[24324] 1631274092.590583: Selected etype info: etype aes256-cts, salt
"SOMEDOMAIN.ORGSomeUser", params ""
[24324] 1631274092.590584: PKINIT loading CA certs and CRLs from FILE
[24324] 1631274092.590585: PKINIT client computed kdc-req-body checksum
9/8C2B5A6C5555AF5D9A19ADFFF88D4114CB5764A1
[24324] 1631274092.590587: PKINIT client making DH request
[24324] 1631274092.590588: Preauth module pkinit (16) (real) returned:
0/Success
[24324] 1631274092.590589: Produced preauth for next request: PA-PK-AS-REQ
(16)
[24324] 1631274092.590590: Sending request (4908 bytes) to SOMEDOMAIN.ORG
[24324] 1631274092.590591: Resolving hostname 10.1.2.3
[24324] 1631274092.590592: Initiating TCP connection to stream 10.1.2.3:88
[24324] 1631274092.590593: Sending TCP request to stream 10.1.2.3:88
[24324] 1631274092.590594: Received answer (5450 bytes) from stream
10.1.2.3:88
[24324] 1631274092.590595: Terminating TCP connection to stream 10.1.2.3:88
[24324] 1631274092.590596: Sending DNS URI query for
_kerberos.SOMEDOMAIN.ORG.
[24324] 1631274092.590597: No URI records found
[24324] 1631274092.590598: Sending DNS SRV query for
_kerberos-master._udp.SOMEDOMAIN.ORG.
[24324] 1631274092.590599: Sending DNS SRV query for
_kerberos-master._tcp.SOMEDOMAIN.ORG.
[24324] 1631274092.590600: No SRV records found
[24324] 1631274092.590601: Response was not from master KDC
[24324] 1631274092.590602: Processing preauth types: PA-PK-AS-REP (17)
[24324] 1631274092.590603: PKINIT client verified DH reply
[24324] 1631274092.590604: PKINIT client config accepts KDC dNSName SAN
10.1.2.3
[24324] 1631274092.590605: PKINIT client config accepts KDC dNSName SAN
10.1.2.4
[24324] 1631274092.590606: PKINIT client config accepts KDC dNSName SAN
10.1.2.3
[24324] 1631274092.590607: PKINIT client config accepts KDC dNSName SAN
10.1.2.4
[24324] 1631274092.590608: PKINIT client found dNSName SAN in KDC cert:
SRV_AD1.adb.somedomain.org
```

```
[24324] 1631274092.590609: PKINIT client found dNSName SAN in KDC cert:
adb.somedomain.org
[24324] 1631274092.590610: PKINIT client found dNSName SAN in KDC cert: ADB
[24324] 1631274092.590611: PKINIT client found no acceptable SAN in KDC cert
[24324] 1631274092.590612: Preauth module pkinit (17) (real) returned:
-1765328308/KDC name mismatch
[24324] 1631274092.590613: Produced preauth for next request: (empty)
[24324] 1631274092.590614: Getting AS key, salt "SOMEDOMAIN.ORGSomeUser",
params ""
kinit: Cannot read password while getting initial credentials
```

Note that the output may be different, but the structure is usually the same.

Rather than focusing on the generic kinit error message, look for the specific error code (the message is likely to be different).

If there is a line such as `-1765328360/Preauthentication failed` or `-1765328254/Cannot read password`: skip this line and continue to the top to find a more specific error code.

kinit: Cannot contact any KDC for realm 'SOMEDOMAIN.ORG' while getting initial credentials

- Received error from KDC: -1765328332/Response too big for UDP, retry with TCP

kinit: Cannot read password while getting initial credentials

- Preauth module pkinit (17) (real) returned: -1765328307/Inconsistent key purpose
- Preauth module pkinit (17) (real) returned: -1765328308/KDC name mismatch
- Preauth module pkinit (17) (real) returned: -1765328313/Failed to verify received certificate (depth 0): certificate has expired
- Preauth module pkinit (17) (real) returned: -1765328314/Failed to verify received certificate (depth 0): unable to get local issuer certificate
- Received error from KDC: -1765328361/Password has expired

kinit: Certificate mismatch while getting initial credentials

- Received error from KDC: -1765328318/Certificate mismatch

kinit: Client not trusted while getting initial credentials

- Received error from KDC: -1765328322/Client not trusted

kinit: Client 'someuser\@somedomain.org@SOMEDOMAIN.ORG' not found in Kerberos database while getting initial credentials

- Received error from KDC: -1765328378/Client not found in Kerberos database

kinit: Client's credentials have been revoked while getting initial credentials

- Received error from KDC: -1765328366/Client's credentials have been revoked

kinit: KDC has no support for padata type while getting initial credentials

- Received error from KDC: -1765328368/KDC has no support for padata type

kinit: Pre-authentication failed: Failed to verify own certificate (depth 1): unable to get issuer certificate

while getting initial credentials

Note: "depth" might have a different value.

- Preauth module pkinit (16) (real) returned: -1765328360/Failed to verify own certificate (depth 1)

Background info on "depth" value:

Depth 0: leaf certificate
Depth 1: issuer of certificate #0. This will either be an intermediate or root CA (only if this is the last certificate in the chain).
Depth 2: in the case below, it's the last certificate in the chain, which is the root CA.

```
Certificate chain
 0 s:/CN=somedomain.org
   i:/C=US/O=Let's Encrypt/CN=R3
 1 s:/C=US/O=Let's Encrypt/CN=R3
   i:/C=US/O=Internet Security Research Group/CN=ISRG Root X1
 2 s:/C=US/O=Internet Security Research Group/CN=ISRG Root X1
   i:/O=Digital Signature Trust Co./CN=DST Root CA X3
```

-
