Databricks Private Link Implementation

In accordance with security design, we have implemented both `backend private link` and `front end private link` for Jemena's databricks workspaces; this page serves to clarify the implementation detail of how each is deployed.

Enable private connectivity using AWS PrivateLink | Databricks on AWS

Back End Private Link

As per network design diagram shown in Infrastructure Design, 2 ENIs (SCC, Rest) are deployed for each `nonprod` and `prod` VPCs and registered within the respective databricks accounts. https://gitlab.com/jemena/projects/future-networks-datahub/databricks/core-network-databricks-vpc-components/-/blob/main/main-npd-vpc-resources.tf?ref_type=heads Connect your GitLab account

Databricks workspaces's data plane on each databricks account use the corresponding registered backend ENIs to communicate with control plane.

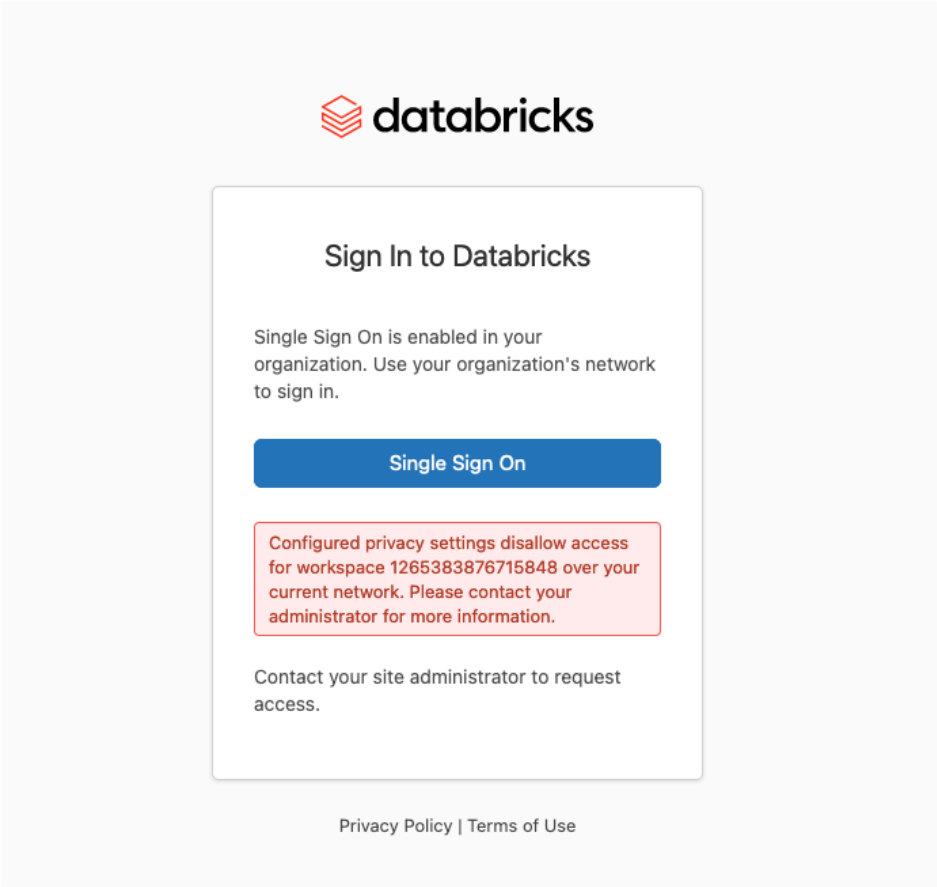
Front End Private Link

Front end private link ENI is only deployed on the `prod` VPC https://gitlab.com/jemena/projects/future-networks-datahub/databricks/core-network-databricks-vpc-components/-/merge_requests/14 Connect your GitLab account ; however, it is used for communication with both `nonprod` and `prod` databricks workspaces. Reason for this is to simplify DNS configuration within Jemena core network when routing traffic to the corresponding workspaces (if we deploy separate FEPL for nonprod and prod, we'd need to specify fully qualified names for each workspace).

Note that it is also possible to use this `FEPL` for `LV DERMs` workspace front end communication without further core network configuration.

Further reading on `FEPL` here: Configuring DNS resolution for Private Databricks Workspaces (AWS)

If user attempts to log into a workspace from public network, the following error will be displayed.



FEPL Connectivity Tests

https://gitlab.com/jemena/projects/future-networks-datahub/databricks/databricks-workspaces/-/merge_requests/43/diffs Connect your GitLab account

```
8459428cd ...
$ terraform version
Terraform v1.9.8
on linux_amd64
$ cd stacks/elec-network-bu
$ ../../scripts/fepl-test.sh
fepl test
fepl test
fepl test
fepl test
fepl test
fepl test
fepl test
fepl test
Server:         10.32.0.2
Address:        10.32.0.2:53
Non-authoritative answer:
dbc-de6c0ca1-0e35.cloud.databricks.com  canonical name = sydney.privatelink.cloud.databricks.com
Name:   sydney.privatelink.cloud.databricks.com
Address: 10.32.78.86
Name:   sydney.privatelink.cloud.databricks.com
Address: 10.32.78.18
Name:   sydney.privatelink.cloud.databricks.com
Address: 10.32.78.59
Non-authoritative answer:
dbc-de6c0ca1-0e35.cloud.databricks.com  canonical name = sydney.privatelink.cloud.databricks.com
fetch https://dl-cdn.alpinelinux.org/alpine/v3.20/main/x86_64/APKINDEX.tar.gz
fetch https://dl-cdn.alpinelinux.org/alpine/v3.20/community/x86_64/APKINDEX.tar.gz
v3.20.3-171-ge0d8a949f52 [https://dl-cdn.alpinelinux.org/alpine/v3.20/main]
v3.20.3-170-gb7926213bd4 [https://dl-cdn.alpinelinux.org/alpine/v3.20/community]
OK: 24163 distinct packages available
(1/1) Installing curl (8.10.1-r0)
Executing busybox-1.36.1-r29.trigger
OK: 26 MiB in 38 packages
Installed Databricks CLI v0.230.0 at /usr/local/bin/databricks.
Databricks CLI v0.230.0
05:31:54  INFO start pid=50 version=0.230.0 args="databricks, clusters, list-zones, --log-level, info"
05:31:55  INFO completed execution pid=50 exit_code=0
{
  "default_zone":"ap-southeast-2c",
  "zones": [
    "ap-southeast-2c",
    "ap-southeast-2a",
    "ap-southeast-2b"
  ]
}
Cleaning up project directory and file based variables
Job succeeded
```