SIRIUS FG LAB · CLOUD ENGINEERING

Azure Build Step-by-Step Guide

A HANDS-ON COURSE · BUILD TO TEARDOWN

Re-platforming the Docker estate onto disposable Azure IaaS
Grounded in CAF · WAF · Bicep + deployment stacks
Authorized training use only · v1 · 2026-10-01

Part I — How this guide works

This is a hands-on course for building the Sirius FG Lab on Azure, written for someone meeting Azure, Infrastructure-as-Code, and AD security for the first time. You will stand up a deliberately-vulnerable Active Directory estate on isolated, disposable cloud infrastructure, validate the attack paths against it, and tear it back down to zero cost. Every command here was fact-checked against live Microsoft documentation.

> **LEARNING OBJECTIVES**
>
> By the end you can: provision an isolated Azure resource group and private network with Bicep; deploy a managed jump host (Bastion) as the single entry point; run a multi-container vulnerable AD estate on a no-public-IP VM; wrap it all in a deployment stack for one-command teardown; and reason about the cost and security trade-offs of a disposable lab.

The four callout types

> **KEY CONCEPT**
>
> Explains the "why" behind a step — the idea you're meant to carry forward.

> **CHECKPOINT**
>
> A concrete thing to verify before moving on, so you never build on a broken step.

> **COMMON PITFALL**
>
> A real mistake that bites first-timers here, and how to avoid it.

> **LEARNING OBJECTIVES**
>
> What a section will let you do once complete.

Prerequisites

- An Azure subscription you can create resources in (a dedicated one is strongly recommended — see the isolation note below).
- The Azure CLI (`az`) on your PATH. On Kali, install via `pipx` (its apt package is broken on Python 3.14).
- Bicep (`az bicep install`) and the `bastion` CLI extension.
- Comfort with a terminal. No prior Azure experience assumed.

> **COMMON PITFALL — ISOLATION FIRST**
>
> A vulnerable domain controller on a public IP is a gift to internet scanners. This lab is built **private-by-default**: no VM gets a public IP, and the one audited exception is Azure Bastion. Use a dedicated subscription or at least a dedicated resource group so the blast radius and the cost boundary are unambiguous.

Big-picture architecture

The cloud estate mirrors the local Docker lab's network (`10.30.1.0/24`) exactly, so the IPs you learned in the live demo still hold:

HOST	IP	ROLE	PLANTED WEAKNESS
dc01	10.30.1.10	Primary DC (Samba 4)	Kerberoastable SPNs; AS-REP user
dc02	10.30.1.11	Replica DC	DCSync target
ca01	10.30.1.12	ADCS (Flask certsrv)	ESC1 template
sql01	10.30.1.40	MS SQL 2019	xp_cmdshell; svc_sql sysadmin
vault01	10.30.1.70	HashiCorp Vault	Seeded prod secrets (the objective)
gateway	10.30.1.254	Status page	—

In the cloud, all six run as Docker containers on a single no-public-IP VM (`vm-lab-estate`), fronted by Azure Bastion inside a VNet with a deny-by-default NSG.

KEY CONCEPT — WHY BICEP + DEPLOYMENT STACKS

The estate is Azure-only and disposable, so we skip Terraform's state file entirely. A **deployment stack** is an Azure-native object that tracks everything it deploys; a single `az stack group delete --action-on-unmanage deleteAll` removes exactly what was created, tearing down to zero with no leftover state to manage.

Part II — Phase 0 · Foundation

OBJECTIVES

Install the toolchain, authenticate, pick the right subscription, create the resource group, and set cost guardrails.

Install & authenticate

```
# verify the CLI and sign in (opens a browser)
az version
az login
az account show -o table        # confirm the ACTIVE subscription
az account set --subscription "<name-or-id>"   # if it's the wrong one
```

COMMON PITFALL — THE WRONG SUBSCRIPTION

Everything you create lands in whatever subscription is active. Confirm it before you deploy anything — the launcher makes this a one-time confirmation gate for exactly this reason.

Resource group + guardrails

```
az group create --name rg-siriusfg-lab --location eastus
# Defender for Servers is optional + paid; enable only if you want it
az security pricing create --name VirtualMachines --tier standard
```

CHECKPOINT

`az group show --name rg-siriusfg-lab` returns the group. If Defender errors with "Not Registered," that's fine — it's optional; skip it.

Phase 1 · Network & the Isolation Gate

OBJECTIVES

Deploy the VNet, subnets, and a deny-by-default NSG with Bicep, then stand up Azure Bastion as the only entry point.

The network, in Bicep

The network lives in `network.bicep`: a VNet at `10.30.0.0/16`, the workload subnet `snet-lab` (10.30.1.0/24) with an attached NSG, and the dedicated `AzureBastionSubnet` (10.30.3.0/26). Deploy it:

```
az deployment group create \
  --resource-group rg-siriusfg-lab \
  --template-file network.bicep \
  --parameters location=eastus
```

COMMON PITFALL — AZUREBASTIONSUBNET SIZING

Bastion requires a subnet named exactly `AzureBastionSubnet`, sized **/26 or larger**. A /27 is rejected. And every Bicep file needs `param location string = resourceGroup().location` — omitting it is a classic first-deploy failure.

Bastion — the one way in

```
az network public-ip create -g rg-siriusfg-lab -n pip-bastion --sku Standard --allocation-method Static
az network bastion create -g rg-siriusfg-lab -n bas-siriusfg \
  --public-ip-address pip-bastion --vnet-name vnet-siriusfg-lab \
  --location eastus --sku Standard --enable-tunneling true
```

KEY CONCEPT — TUNNELING

`--enable-tunneling true` is what lets you run `az network bastion ssh` and open native tunnels to a private VM. Without it you'd be limited to the browser console. Bastion takes 10–15 minutes to deploy — that silence is normal.

ISOLATION GATE

Before deploying anything vulnerable: confirm **no resource has a public IP except `pip-bastion`**, and the NSG denies inbound from the internet. This gate is the whole safety model — do not pass it until it's true.

Phase 2 · The estate VM

OBJECTIVES

Deploy a no-public-IP VM, have cloud-init install Docker, reach it over Bastion, bring up the estate, and validate the five techniques.

The VM (no public IP)

```
az vm create -g rg-siriusfg-lab -n vm-lab-estate \
  --image Ubuntu2204 --size Standard_D2s_v4 \
  --vnet-name vnet-siriusfg-lab --subnet snet-lab \
  --nsg "" --public-ip-address "" \
  --admin-username azureuser --generate-ssh-keys \
  --custom-data cloud-init-docker.yaml
```

COMMON PITFALL — VM SIZE & ARCHITECTURE

Whole VM families can be NotAvailableForSubscription in a region (the entire B-series was blocked in eastus at build time — hence `D2s_v4`). And never pick an ARM (p-series) size: the MSSQL container has no ARM build and will fail to start.

Reach it and bring the estate up

```
az network bastion ssh -n bas-siriusfg -g rg-siriusfg-lab \
  --target-resource-id $(az vm show -g rg-siriusfg-lab -n vm-lab-estate --query id -o tsv) \
  --auth-type ssh-key --username azureuser --ssh-key ~/.ssh/id_rsa
# on the VM: confirm cloud-init finished, then start the estate
cloud-init status --wait
sudo docker compose -f siriusfg-lab/docker-compose.yml up -d
```

KEY CONCEPT — MOVING FILES OVER BASTION

To copy the estate onto the VM, use `az network bastion tunnel` to open a local port forward and `scp` through it — there is no direct public SSH target. (A fabricated direct `scp` target is a common copy-paste error; the tunnel is the real path.)

CHECKPOINT — VALIDATE THE FIVE TECHNIQUES

From a Kali box tunneled in, confirm each path works against the live estate: AS-REP roast (`asmith`), Kerberoast (`svc_sql` / `svc_web`), GPP cpassword decrypt, `mssqlclient` + `xp_cmdshell` on sql01, and certipy ESC1 against ca01.

Phase 3 · Deployment stack & the Cost Gate

OBJECTIVES

Wrap the estate in a deployment stack, learn the build→destroy lifecycle, and confirm teardown returns to $0.

```
az stack group create --name siriusfg-lab-stack \
  --resource-group rg-siriusfg-lab \
  --template-file main.bicep \
  --action-on-unmanage deleteAll --deny-settings-mode none --yes
```

KEY CONCEPT — DESTROY, DON'T STOP

A deallocated VM still bills for its disks. A destroyed resource group bills nothing. The entire cost model of this lab is "spin up, use, tear all the way down." Bastion Standard (~$0.29/hr) is the dominant cost, so leaving it idle overnight is the expensive mistake.

Teardown

```
az stack group delete --name siriusfg-lab-stack -g rg-siriusfg-lab \
  --action-on-unmanage deleteAll --yes
# reliable fallback that always stops billing:
az group delete --name rg-siriusfg-lab --yes
az group exists --name rg-siriusfg-lab   # expect: false
```

COMMON PITFALL — STACK OWNERSHIP

If the stack only tracks the network (because the Bastion/VM were made by direct CLI), `deleteAll` can fail trying to delete the VNet while those unowned resources still sit in its subnets. The launcher handles this by falling back to a resource-group delete; the `reference-full-stack` Bicep set makes the stack own everything so the primary path succeeds cleanly.

COST GATE

Teardown isn't done until `az group exists` prints `false` (or the resource group is gone from the portal). That is the only reliable "$0" signal — Azure's cost dashboard lags ~24h.

Phase 4 · Optional tracks

- **Go native** — replace the Samba/Flask simulation with real Windows AD + AD CS once a working baseline exists.
- **Demo button** — wire a "LAUNCH LAB" button into the live demo that points at the running estate.
- **Steps 5–7** — extend the attack chain: lateral movement, persistence (krbtgt DCSync, shadow creds), exfiltration (share harvest, Vault dump).

Phase 5 · Department Tier (optional, independent)

OBJECTIVES

Add HR/IT/Finance child domains with their own file shares and domain-joined clients — a realistic multi-domain forest for cross-domain attacks. ~12 hours, independent of the core timeline.

This is a seven-VM expansion (3 child-domain DCs + 4 clients), covered end to end in the companion Department Expansion Build Guide. In brief: host the two provisioning PowerShell scripts at a public HTTPS URL, then deploy one orchestrator template:

```
az deployment group create -g rg-siriusfg-lab \
  --template-file dept-expansion/bicep/main-departments.bicep \
  --parameters adminPassword='<...>' rootAdminPassword='<...>' \
               scriptBaseUri='https://<host>/scripts'
```

KEY CONCEPT — CHILD DOMAINS & UNIVERSAL GROUPS

Each department is a real child domain of `SIRIUSFG.LOCAL` with an automatic two-way transitive trust. The `IT-Support` group is **Universal** scope so it can be referenced in ACLs across the HR and Finance domains — the intended cross-domain escalation vector.

CHECKPOINT

From dc01: `Get-ADForest` lists hr/it/fin; `Get-ADTrust` shows each as BiDirectional.

Part III — Reference

Command reference

TASK	COMMAND
Sign in / check account	`az login` · `az account show -o table`
Create resource group	`az group create -n rg-siriusfg-lab -l eastus`
Deploy a Bicep file	`az deployment group create -g <rg> --template-file x.bicep`
Create Bastion	`az network bastion create ... --enable-tunneling true`
SSH via Bastion	`az network bastion ssh ...`
Tunnel a port via Bastion	`az network bastion tunnel ...`
Create a VM (no public IP)	`az vm create ... --public-ip-address "" --nsg ""`
Create deployment stack	`az stack group create ... --action-on-unmanage deleteAll`
Tear down (stack)	`az stack group delete ... --action-on-unmanage deleteAll --yes`
Tear down (nuclear)	`az group delete -n rg-siriusfg-lab --yes`
Confirm $0	`az group exists -n rg-siriusfg-lab` → `false`

Glossary

TERM	MEANING
Bastion	Managed jump host; brokers SSH/RDP into private VMs with no exposed ports.
Deployment stack	Azure object tracking a set of resources so they delete together cleanly.
NSG	Network Security Group — subnet/NIC firewall rules; here, deny-by-default.
cloud-init	First-boot config that installs Docker on the estate VM.
Child domain	A subdomain of the forest root with an automatic two-way transitive trust.
Universal group	Group scope that can be referenced across all domains in the forest.

Troubleshooting by phase

SYMPTOM	CAUSE / FIX
`az` crashes: 'NoneType' object is not callable	Kali's apt azure-cli on Python 3.14. Install via `pipx` instead.
Bastion create rejected	AzureBastionSubnet too small — needs /26+.

VM create: SkuNotAvailable	Family blocked for the subscription/region. Use `D2s_v4` (x64).
Teardown: DeploymentStackDeleteResourcesFailed	Stack doesn't own Bastion/VM. Fall back to `az group delete`.
Still billing after "stop"	Deallocated VMs bill for disks. Destroy, don't stop.

Every Azure CLI/Bicep command here was fact-checked against live Microsoft docs. Companion documents: the Department Expansion Build Guide and the Azure Overview PM deck. Source of truth: github.com/nagini-x/siriusfg-lab.