CETTABYTE
TECHNICAL WHITE PAPER

Hardening the Azure Perimeter and Workload Plane

A defense-in-depth reference for network, compute and workload security across Azure Commercial and Azure Government

Cettabyte | Cloud Architecture, Software and AI
Written by Lina Brihoum
July 2026

Contents

1. Executive Summary **3**

2. Scope, Method and Conventions **4**
2.1 In scope 4
2.2 Out of scope 4
2.3 Conventions 4

3. Threat Model: Why Defaults Are Not a Baseline **4**
3.1 The canonical attack path 5
3.2 Four structural failure modes 5

4. Design Principles **6**

5. Part I — Network and Perimeter Hardening **8**
5.1 Topology and segmentation 8
5.2 Network security groups and application security groups 9
5.3 Egress architecture 10
5.4 Azure Firewall 11
5.5 Private Link and private endpoints 13
5.6 Web application firewalling 14
5.7 DDoS protection 15
5.8 Network-plane telemetry 16

6. Part II — Compute and Workload Hardening **17**
6.1 Platform integrity for virtual machines 17
6.2 Image supply chain 17
6.3 Encryption at rest 18
6.4 Administrative access paths 18
6.5 Patch and configuration management 19
6.6 Azure Kubernetes Service 19
6.7 App Service and Azure Functions 21
6.8 Secrets and key management 22

7. Part III — Enforcement: Policy as the Control Plane **24**
7.1 Effects and where each belongs 24
7.2 A rollout pattern that does not cause outages 24
7.3 Initiatives and frameworks 24
7.4 What policy cannot do 25

8. Part IV — Azure Government Deltas **26**
8.1 It is a different cloud, not a different region 26
8.2 Compliance posture 26
8.3 Service availability and feature lag 27
8.4 Operational differences that surface late 27

9. Implementation Roadmap **28**
9.1 A note on sequencing 29

Appendix A — Hardening Checklist **30**

About Cettabyte **32**

1. Executive Summary

Microsoft operates a demonstrably secure platform. That fact is frequently mistaken for a guarantee that workloads running on it are secure. They are not, and the distinction is where nearly every incident we investigate actually lives. In the shared responsibility model, Microsoft secures the hypervisor, the physical estate, the control plane and the service fabric. Everything above that boundary — network reachability, identity scope, image provenance, egress control, key custody — is the customer's to configure. Azure ships with permissive defaults chosen for onboarding velocity, not for adversary resistance.

This paper concerns two of those planes: the **network and perimeter** layer, and the **compute and workload** layer. We treat them together deliberately. Perimeter controls without workload integrity produce a hard shell around a soft interior; workload hardening without egress control produces a well-built machine that still calls out to an attacker-controlled endpoint. The two only work as a pair.

Four conclusions run through everything that follows.

1. **Egress is the control you are most likely missing.** Inbound filtering is well understood and usually present. Outbound filtering is frequently absent, and it is the path that command-and-control, token exfiltration and data theft all traverse. Azure's recent shift to private-by-default subnets removes the implicit egress path for new virtual networks, which is an opportunity to redesign rather than a problem to work around.
2. **Private Link is a DNS project that happens to involve networking.** Most failed or partially effective private endpoint deployments we encounter fail at name resolution, not at connectivity — and a private endpoint that coexists with an enabled public data plane provides the appearance of isolation without the substance.
3. **Workload identity has replaced secrets, but most estates have not finished migrating.** Federated, short-lived, workload-scoped credentials eliminate whole categories of exposure. Long-lived service principal secrets and connection strings persist in application settings, pipeline variables and Kubernetes secret objects because nobody owns the migration.
4. **Policy, not documentation, is the enforcement boundary.** A hardening standard that exists only as a wiki page decays within one quarter. The same standard expressed as Azure Policy with deny and deployIfNotExists effects survives staff turnover, holds under deployment pressure, and generates its own audit evidence.

The controls described here are available to any Azure subscription today. Almost none of them require a premium tier that an enterprise does not already hold. The gap between an average Azure estate and a hardened one is, with few exceptions, a matter of configuration discipline and enforcement architecture rather than budget or capability.

> **WHO THIS PAPER IS FOR**
>
> Cloud platform and infrastructure engineers responsible for landing zone design and implementation; security architects reviewing an Azure design against a control framework; and technology leaders who need to

> understand what "hardened" concretely means before funding it. Sections are written to be read independently — a network engineer can start at Part I and a Kubernetes team at §6.6 without loss of context.

2. Scope, Method and Conventions

We organise this paper around controls rather than products. Azure services are renamed, merged and superseded on a cadence that outpaces most documentation; the underlying control objectives — restrict reachability, verify integrity, scope credentials, prove state — do not change. Where a specific service is the practical implementation of a control, we name it and note its constraints.

2.1 In scope

- Network and perimeter: virtual network topology and segmentation, network security groups, egress architecture, Azure Firewall, Private Link and private DNS, web application firewalling, DDoS protection, and network-plane telemetry.
- Compute and workload: virtual machine platform integrity and image supply chain, encryption at rest, administrative access paths, patch and configuration management, Azure Kubernetes Service, App Service and Functions, and secrets and key management.
- Enforcement: Azure Policy as a control plane, infrastructure-as-code guardrails, and compliance evidence generation.
- Azure Government: architectural, endpoint and compliance deltas from Azure Commercial, called out inline and consolidated in Part IV.

2.2 Out of scope

Identity and governance are treated only where they intersect the two planes above — managed identity scoping, Privileged Identity Management for administrative paths, Conditional Access on the management plane. A full treatment of Microsoft Entra ID hardening, data platform security (SQL, Cosmos DB, Fabric), security operations engineering, and incident response is deliberately excluded; each warrants its own paper.

2.3 Conventions

- **Gov Delta** callouts mark behaviour that differs materially in Azure Government. Absence of a callout should not be read as a guarantee of parity — always verify current regional availability before committing to a design.
- **Control statements** are written as imperatives. Where a control has a meaningful cost, operational burden or compatibility risk, we say so rather than presenting it as free.
- Retirement and deprecation dates were current as of publication. Azure lifecycle dates move — the *direction* of a deprecation is far more durable than its announced date, and should drive design.

3. Threat Model: Why Defaults Are Not a Baseline

Hardening without a threat model produces checklists that are simultaneously exhausting and incomplete. The model below is deliberately narrow: it describes how attackers actually move through Azure estates, which is rarely through a platform vulnerability and almost always through customer configuration.

3.1 The canonical attack path

A representative intrusion follows five stages. Each stage maps to a control in this paper, and each stage is a place the chain can be broken.

Stage	What the attacker does	Enabling misconfiguration	Primary control
Initial access	Exploits an internet-reachable application, or authenticates to an exposed management port with credentials obtained elsewhere.	RDP/SSH open to Internet; VM with a public IP; unfiltered web tier with no WAF.	No public IPs on compute; Bastion + JIT; WAF in prevention mode (§5.6, §6.4)
Execution and discovery	Runs code in the workload context; enumerates the subscription via the resource graph and the local metadata service.	Over-permissioned managed identity; no runtime detection on the host or cluster.	Least-privilege identity scope; Defender for Servers / Containers (§6.5, §6.6)
Credential access	Retrieves a managed identity access token from the Instance Metadata Service at 169.254.169.254, frequently via server-side request forgery in the application itself.	Application vulnerability plus a broadly-scoped identity; no IMDS access restriction.	Scope identities per-workload; restrict IMDS reachability; WAF SSRF rules (§6.8)
Lateral movement	Moves east–west to higher-value subnets and peered networks using the stolen token or reused credentials.	Flat VNet — the default AllowVnetInBound rule permits all intra-VNet traffic; hub peering with forwarded traffic enabled.	Explicit intra-VNet deny; ASG-based microsegmentation; network policy in AKS (§5.2)
Exfiltration	Stages data and moves it to attacker-controlled infrastructure over an unfiltered outbound path.	Implicit outbound access; no FQDN filtering; public PaaS data planes reachable from the workload.	Forced tunnelling to a firewall; FQDN and IDPS filtering; private endpoints with public access disabled (§5.3, §5.5)

3.2 Four structural failure modes

Beneath the individual misconfigurations sit four recurring structural problems. They are worth naming because remediating instances without addressing the structure guarantees recurrence.

Implicit egress

For most of Azure's history, a virtual machine deployed into a virtual network with no explicit outbound configuration received internet access anyway, via a platform-assigned public IP address the customer neither chose nor controlled. This is the **default outbound access** behaviour. It bypasses content filtering, produces no egress logs the customer owns, and uses an address that Microsoft can change without notice.

Microsoft has now reversed this default. For virtual networks created with API versions released after **31 March 2026**, the `defaultOutboundAccess` property on subnets defaults to `false` — new VNets are private by default and require an explicit outbound method. Existing virtual networks and subnets are unaffected. This is the single most consequential Azure networking default change in years, and it is worth treating as a forcing function for a deliberate egress design rather than a compatibility problem to be silenced.

Two practical notes. First, the Azure portal adopts new API versions before infrastructure-as-code that pins older ones, so portal-created and pipeline-created networks may behave differently during the transition. Second, a documented platform exception exists: virtual machines in a private subnet can still reach Azure Storage accounts in the same region without an explicit outbound method. If your control objective is that all egress transits an inspection point, that path must be closed with network security group rules.

Flat networks

Every network security group carries a default rule, `AllowVnetInBound` at priority 65000, permitting all traffic sourced from the `VirtualNetwork` service tag. That tag covers the entire address space of the VNet, all peered VNets, and connected on-premises ranges. The practical consequence is that an Azure estate is, absent explicit rules, laterally flat by default — an attacker with a foothold in a low-value subnet has unimpeded network access to every other subnet in the topology. Introducing an explicit intra-VNet deny beneath the default rules, with narrowly scoped allows above it, is the highest-value single change most organisations can make to their network configuration.

Over-permissioned workload identity

Managed identity removed the need to store credentials, which was an unambiguous improvement. It did not remove the need to scope them. We routinely find identities assigned Contributor at subscription scope because that was the fastest way to make a deployment succeed. When such an identity's token is stolen from the metadata endpoint, the blast radius is the subscription. Identity scope is now the dominant determinant of blast radius in cloud environments — more so than network position.

Unmanaged image and configuration drift

A virtual machine hardened by hand at build time diverges from that baseline within weeks. Without a versioned image pipeline and continuous configuration assessment, "hardened" describes a moment in the past rather than a current state. The same applies to container images: an image scanned once at build and never re-evaluated accumulates known vulnerabilities silently.

4. Design Principles

The recommendations in Parts I and II derive from six principles. Where a specific control seems onerous, it is usually worth returning to the principle it serves and asking whether a cheaper control satisfies the same objective.

5. **Deny by default at every plane.** Network, identity, and admission control each need an explicit deny floor with narrowly scoped exceptions above it. A plane that defaults to allow will accumulate implicit permissions that nobody can enumerate.
6. **Egress is explicit and inspected.** Every outbound path should traverse a control point that can filter by destination, log the flow, and be reasoned about in an audit. "The workload needs internet access" is a requirement to be decomposed into named destinations, not a reason to open 0.0.0.0/0.
7. **Identity is the data-plane perimeter; the network is the blast-radius boundary.** PaaS data planes are reached over Microsoft's backbone and authorised by identity, not by IP address. Network controls remain essential — but their job is to contain a compromise, not to authenticate one.
8. **Compute is immutable and attested.** Instances are built from versioned, scanned, signed images and replaced rather than patched in place. Boot integrity is measured and verified by the platform, not asserted by the operator.
9. **Enforcement lives in code.** Controls are expressed as policy definitions and pipeline gates, versioned in source control, and reviewed like any other change. Human review is a supplement to automated enforcement, never a substitute.
10. **Design for the audit.** Every control should emit evidence as a byproduct of operating. If demonstrating compliance requires a manual screenshot exercise, the control will not be demonstrated reliably, and the design has a gap regardless of its technical merit.

5. Part I — Network and Perimeter Hardening

5.1 Topology and segmentation

Topology decisions constrain every subsequent control, and they are expensive to revisit. Two patterns dominate: a customer-managed hub-and-spoke, and Azure Virtual WAN, in which Microsoft manages the transit hub.

Dimension	Hub-and-spoke (customer-managed)	Virtual WAN (Microsoft-managed)
Transit control	Full. Routing, NVA placement and traffic steering are explicit and inspectable.	Abstracted. Routing intent and policies replace hand-authored route tables.
Best fit	Regional estates, strict route-control requirements, third-party NVAs with unusual topologies.	Many regions, many branch sites, large-scale any-to-any connectivity.
Operational burden	Higher — user-defined routes, peering meshes and gateway transit are managed by you.	Lower for connectivity; the abstraction can obscure the effective route table during incidents.
Inspection	Azure Firewall or NVA in the hub, reached by UDR from every spoke.	Secured virtual hub with Azure Firewall or a supported partner appliance; routing intent handles steering.

Whichever pattern is chosen, three segmentation rules hold.

- **Segment by trust boundary and blast radius, not by application.** A subnet per application produces hundreds of subnets and a rule set nobody maintains. Segment by tier and sensitivity — presentation, application, data, management — and use application security groups for intra-tier distinctions.
- **Reserve platform subnets correctly and generously.** AzureFirewallSubnet, AzureBastionSubnet and GatewaySubnet have fixed names and minimum sizes; Azure Firewall requires a /26 and its management subnet, where used, another. Undersizing these blocks later SKU changes and forces redeployment.
- **Treat peering as a permission grant.** Enable gateway transit and forwarded-traffic allowances only where a routing requirement demands them. A spoke that can forward traffic becomes a transit path between other spokes.

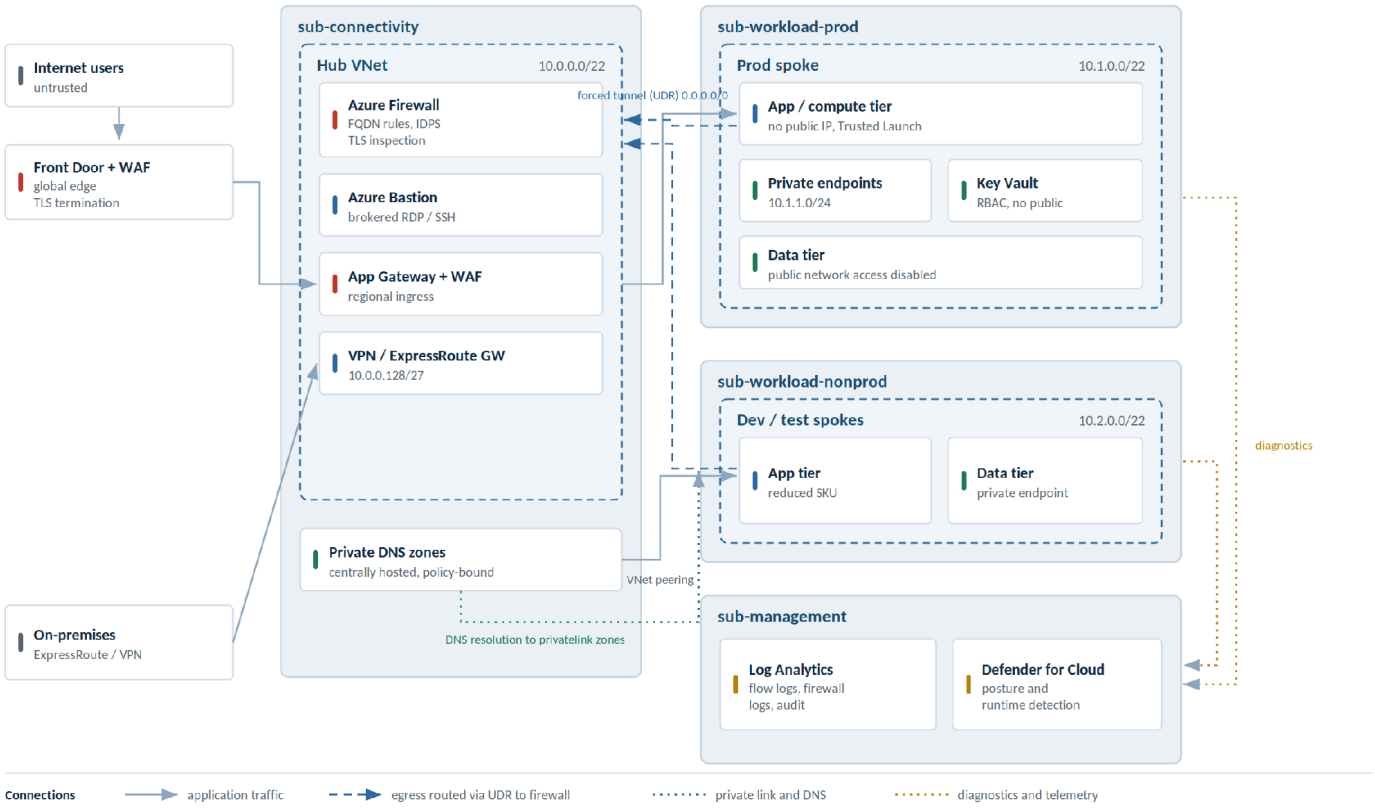

Figure 1 — Reference topology. Ingress is consolidated behind WAF-enforced entry points, all spoke egress is forced through the hub firewall by user-defined route, PaaS data planes are reached only over private endpoints, and telemetry lands in a separate management subscription.

5.2 Network security groups and application security groups

Network security groups are stateful, five-tuple filters evaluated at subnet and network-interface scope. They are the cheapest segmentation control in Azure and the most commonly misconfigured.

Default rules and the intra-VNet problem

Three inbound and three outbound default rules exist at priorities 65000–65500 and cannot be deleted, only overridden by higher-priority (numerically lower) rules.

Priority	Rule	Effect	Hardening implication
65000	AllowVnetInBound / AllowVnetOutBound	Permits all traffic to and from the VirtualNetwork service tag.	The source of network flatness. Override with an explicit intra-VNet deny and scoped allows above it.
65001	AllowAzureLoadBalancerInBound	Permits health probes from 168.63.129.16.	Retain. Removing it silently breaks load balancer and platform health probing.

Priority	Rule	Effect	Hardening implication
65001	AllowInternetOutBound	Permits all outbound traffic to the Internet service tag.	Override with a deny and route egress through a firewall (§5.3).
65500	DenyAllInBound / DenyAllOutBound	Final deny.	The intended floor — but only reached if the 65000/65001 rules are overridden.

Rule design that survives contact with operations

- **Use a priority band scheme.** Reserve 100–999 for platform and management rules, 1000–3999 for application rules, and 4000+ for the explicit deny floor. Without a scheme, priority collisions accumulate and rule intent becomes unreadable.
- **Use application security groups instead of IP literals.** An ASG is a named group of network interfaces referenced as a rule source or destination. Rules written against ASGs remain correct as instances scale and addresses change; rules written against CIDR blocks do not.
- **Prefer regional service tags.** Where a rule must permit access to an Azure service, use the narrowest available tag — `Storage.eastus` rather than `Storage`, and `Storage` rather than `AzureCloud`. Broad tags authorise every tenant's instance of that service, which is an exfiltration path.
- **Apply at subnet scope by default.** Attaching NSGs at both subnet and NIC means traffic is evaluated twice with independent rule sets, which is a reliable source of intermittent, hard-to-diagnose connectivity failures. Use NIC-level NSGs only where a genuine per-instance exception exists.
- **Never allow Any/Any.** A rule permitting all protocols from all sources should fail a pull request automatically. Enforce with policy, not review.

Flow visibility

Network security group flow logs are being retired on **30 September 2027**, and new NSG flow logs could no longer be created after 30 June 2025. The successor is **virtual network flow logs**, which attach at VNet, subnet or interface scope with no dependency on an NSG being present — closing a real gap, since traffic in subnets without an NSG was previously invisible. Existing NSG flow log records remain in storage under their retention policies after retirement, but the flow log resources themselves are deleted. Organisations still on NSG flow logs should migrate now; Microsoft publishes both a migration script and a built-in policy for VNet-scope enablement.

5.3 Egress architecture

Egress control is the discipline that most reliably distinguishes a hardened estate from a merely well-configured one. Four explicit outbound methods exist, and they are not interchangeable.

Method	Characteristics	Appropriate use
NAT Gateway	Fully managed SNAT with a large, predictable port budget per attached public IP and stable	Default for workloads that need reliable outbound reachability where filtering is

Method	Characteristics	Appropriate use
	egress addressing. No filtering, no application-layer visibility.	provided elsewhere, or where destination allow-listing is not required.
Azure Firewall (via UDR)	Centralised inspection: network rules, FQDN-based application rules, threat intelligence, IDPS and optional TLS inspection. Adds cost and a routing dependency.	Any environment with a destination allow-listing requirement, regulatory egress control, or a need for egress audit evidence.
Load balancer outbound rules	SNAT through a Standard Load Balancer front-end with manual port allocation control.	Existing load-balanced backends where port allocation must be tuned per backend pool.
Instance-level public IP	Direct, uninspected internet path attached to the instance.	Effectively never in a hardened estate. Deny by policy.

The reference pattern

For regulated and enterprise workloads we recommend forced tunnelling: a user-defined route in every spoke subnet sending 0.0.0.0/0 to the hub firewall's private address, with the firewall performing FQDN filtering, threat intelligence enforcement and IDPS. The design has three failure modes worth engineering around in advance.

- **Asymmetric routing.** Traffic entering via a public load balancer or Application Gateway and leaving via the firewall will be dropped. Exclude the relevant source prefixes from the default route, or terminate ingress in the same inspection path.
- **SNAT port exhaustion.** Every SNAT device has a finite port budget per public IP. High-connection-rate workloads — particularly those making many short-lived outbound HTTPS connections — exhaust it and fail intermittently. Size public IP allocation against measured connection rates, use a public IP prefix for predictable scaling, and alert on SNAT port utilisation before it becomes an incident.
- **Platform dependency reachability.** Agents, extensions, activation, update services and telemetry endpoints all require named outbound destinations. Build the allow-list from the service tags and FQDN tags Microsoft publishes rather than discovering each dependency through an outage.

> **DESIGN NOTE — THE PRIVATE SUBNET TRANSITION**
>
> Because new virtual networks created after 31 March 2026 default to private subnets, greenfield deployments now fail closed rather than silently obtaining uncontrolled internet access. Treat this as the moment to establish the egress standard: every new spoke gets a route table with a default route to the inspection point, provisioned by policy at subnet creation rather than added by hand afterwards.

5.4 Azure Firewall

Azure Firewall is a managed, scaling, stateful firewall with three SKUs whose differences are security-relevant rather than merely commercial.

Capability	Basic	Standard	Premium
Network and application rules, FQDN filtering	Yes	Yes	Yes
Threat intelligence-based filtering	Alert only	Alert and deny	Alert and deny
IDPS (signature-based intrusion detection and prevention)	No	No	Yes
TLS inspection	No	No	Yes
URL filtering and web category filtering	No	Category only	Full URL path filtering
Typical fit	Small environments, non-production	General enterprise egress control	Regulated workloads; environments requiring inspection of encrypted egress

Rule processing order

Misunderstanding rule precedence produces rule sets that do not behave as written. Azure Firewall processes DNAT rules first, then network rules, then application rules. Network rules take precedence over application rules — a broad network rule permitting TCP/443 to any destination renders every FQDN-based application rule beneath it decorative. If FQDN filtering is the objective, network rules for the same ports and destinations must be tightly constrained or absent.

Additionally, FQDN filtering *within network rules* requires DNS proxy to be enabled on the firewall, so that the firewall resolves names itself and can correlate resolved addresses to rules. Application rules perform FQDN matching from the HTTP Host header or the TLS SNI field and do not carry this dependency.

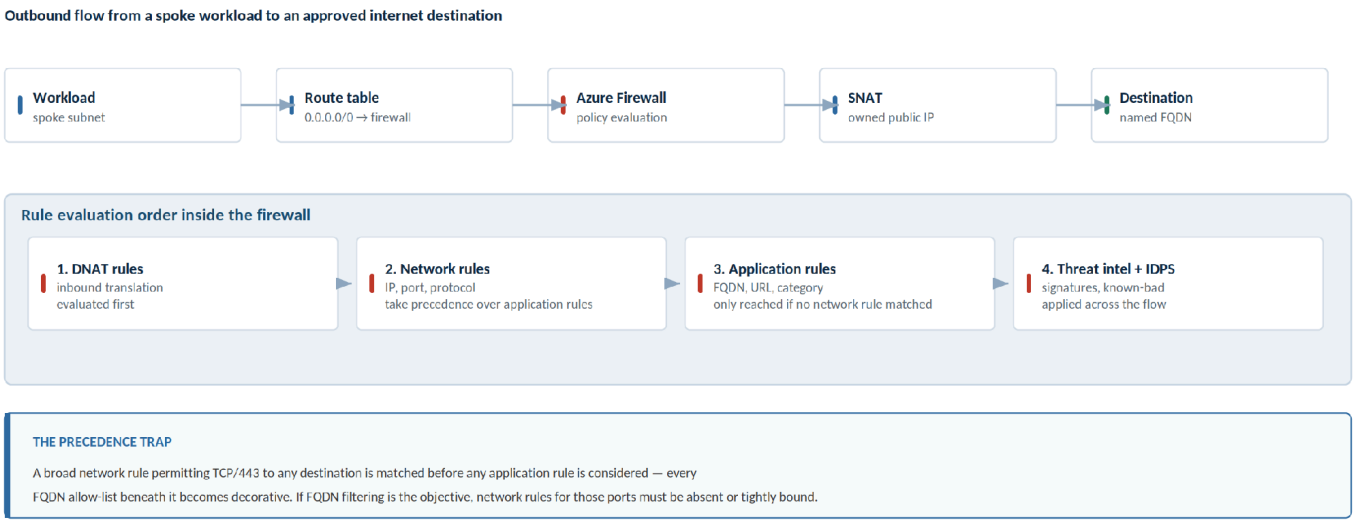

Figure 2 — Egress inspection path and firewall rule precedence. Network rules are evaluated before application rules, which is the mechanism by which a broad port-based allow silently disables an FQDN allow-list.

Policy structure and operational hygiene

- Use Firewall Policy with a parent/child hierarchy: a platform-owned parent policy carrying organisation-wide denies and mandatory allows, inherited by application-team child policies. Parent rule collection groups are evaluated before child ones, which lets the platform team set a floor that application teams cannot override.
- Enable IDPS in alert mode first, baseline the signature noise against real traffic, tune with signature-level overrides, then move to alert-and-deny. Deploying straight to deny reliably breaks production.
- TLS inspection requires an intermediate CA certificate held in Key Vault and accessed by a user-assigned managed identity, plus distribution of the root to every client trust store. It cannot inspect sessions using mutual TLS or certificate pinning — plan explicit bypass lists for those flows rather than discovering them as failures.
- Enable structured, resource-specific diagnostic logs rather than the legacy Azure Diagnostics table. Query performance and retention cost differ substantially at scale, and the structured schema is what current detection content expects.

5.5 Private Link and private endpoints

Private endpoints project a PaaS resource into your virtual network as a network interface with a private address, reached over the Microsoft backbone. They are the correct control for removing PaaS data planes from the public internet — and they are the control most often deployed incompletely.

Private endpoints versus service endpoints

Dimension	Service endpoint	Private endpoint
Addressing	Traffic keeps a public destination address; the source is optimised onto the backbone.	Resource is reachable at a private address inside your VNet.
Cross-premises reach	Not reachable from on-premises over ExpressRoute or VPN.	Reachable from on-premises through the VNet.
Granularity	Service-wide for the subnet.	Per-resource, and per sub-resource (blob, file, table, queue are distinct endpoints).
Exfiltration posture	Permits reaching any account of that service type, including other tenants'.	Reaches only the specific linked resource.
Cost	No charge.	Per-endpoint and per-gigabyte charges.

> **THE MOST COMMON PRIVATE ENDPOINT MISTAKE**
>
> Deploying a private endpoint does not disable the public data plane. Both paths remain open until publicNetworkAccess is explicitly set to Disabled on the resource. An estate can be fully covered by private endpoints and still expose every storage account, key vault and SQL server to the internet. Audit for the resource property, not for the presence of the endpoint.

DNS is the hard part

When a private endpoint is created, the resource's public name is CNAMEd to a name in a `privatelink` subdomain. Resolution then depends entirely on whether the querying client can reach a private DNS zone containing the A record. Get this wrong and clients resolve the public address and either fail closed at the firewall or succeed over the public path — both outcomes look like a working configuration from the resource's side.

A workable enterprise pattern has four parts:

11. Host private DNS zones centrally, in a platform subscription, linked to the hub virtual network. Delegate no zone management to application teams.
12. Enforce automatic A-record registration with a `deployIfNotExists` policy that binds every new private endpoint to the correct zone. Manual registration does not scale and does not survive redeployment.
13. Provide resolution to spokes via DNS Private Resolver or a hub-hosted forwarder, and configure conditional forwarders from on-premises DNS for each privatelink zone.
14. Create endpoints for every sub-resource actually in use. A storage account accessed for both blob and Data Lake operations needs endpoints and zones for both; a missing sub-resource fails only for that access pattern, often long after deployment.

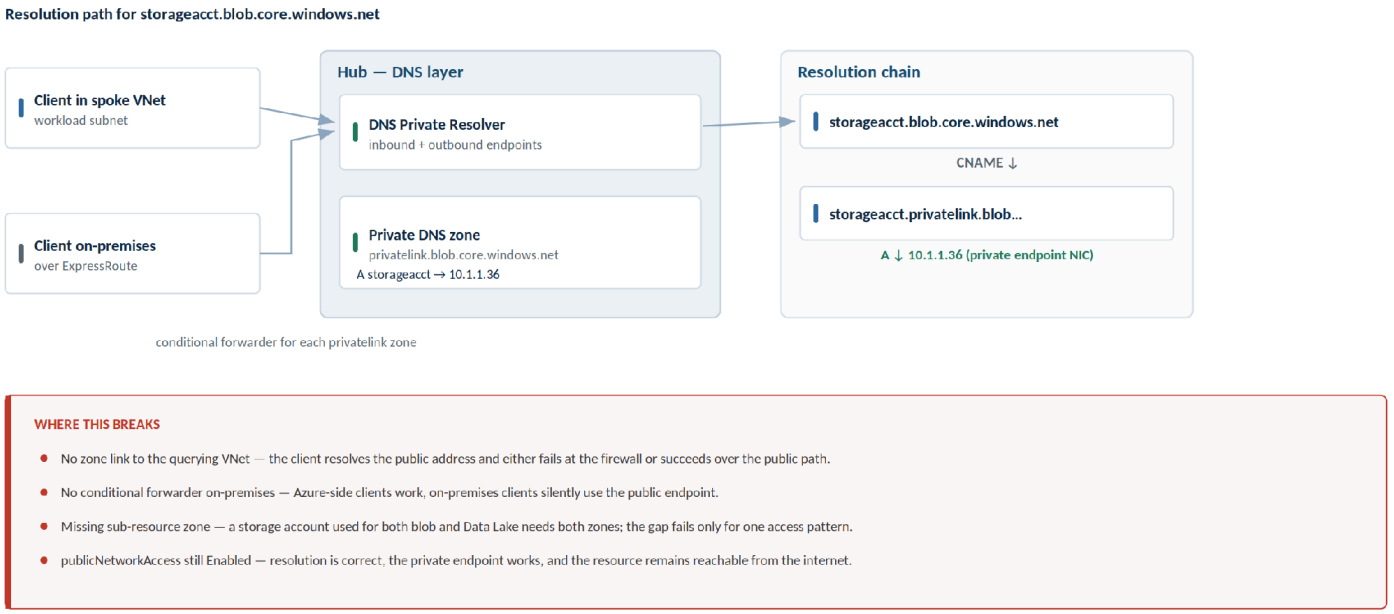

Figure 3 — Private endpoint name resolution, and the four points at which it commonly fails. Each failure mode produces a configuration that appears correct from the resource side.

Private Link as an exfiltration path

Private Link is bidirectional in a sense that is easy to overlook: a private endpoint created in your virtual network can point at a resource in an entirely different tenant, including an attacker-controlled one. The resulting flow leaves over the Microsoft backbone, bypasses your firewall, and does not appear in egress logs. Governance is the only control — use Azure Policy to restrict private endpoint creation to approved resource

types and approved target subscriptions, and alert on private endpoint connection approvals in resources you do not own.

5.6 Web application firewalling

Azure offers WAF in two placements with genuinely different properties.

	Application Gateway WAF v2	Front Door WAF
Placement	Regional, inside your virtual network.	Global, at Microsoft edge points of presence.
Strength	Private backends, mutual TLS termination, fine-grained regional routing.	Edge termination, global anycast, absorbs volumetric traffic before it reaches your region.
Best fit	Internal and regional applications; workloads requiring in-VNet inspection.	Internet-facing global applications; layered in front of a regional gateway.

Getting a WAF to actually block

A WAF left in detection mode is a logging appliance. The reason most stay there is that the transition to prevention is done without a tuning phase and breaks the application. The reliable sequence:

15. Deploy with the current default rule set in detection mode and collect at least one full business cycle of traffic, including batch and month-end patterns.
16. Analyse matches by rule ID and request attribute. Distinguish genuine false positives from real malicious traffic that happens to be constant background noise.
17. **Write narrow exclusions.** Exclude a specific rule for a specific request attribute — a named header, cookie or argument. Disabling an entire rule group because one field trips it removes protection across the whole application.
18. Move to prevention mode, and alert on blocked-request rate so that a tuning regression surfaces as a signal rather than a support ticket.

Beyond the rule set

- Add rate limiting on authentication and other expensive endpoints. Credential stuffing is not caught by signature rules, and rate limits are the cheapest effective control against it.
- Enable managed bot protection and treat verified-bot, known-bad-bot and unknown-bot categories distinctly rather than blocking all automation.
- **Lock down the origin.** A WAF is trivially bypassed if the backend accepts traffic directly. Behind Front Door, restrict the origin to the AzureFrontDoor.Backend service tag *and* validate the X-Azure-FDID header — the service tag alone permits any Front Door profile, including an attacker's. Behind Application Gateway, place the backend on a private endpoint or restrict it to the gateway's subnet.
- Write custom rules for application-specific abuse: geo-filtering where the user population is bounded, and blocks on paths that should never be reachable from the internet.

5.7 DDoS protection

Azure applies basic platform-level DDoS mitigation to all traffic at no charge. It protects the platform, not your application, and it does not tune to your traffic profile. DDoS Network Protection adds adaptive tuning based on learned traffic patterns, attack telemetry and mitigation reports, rapid-response support engagement during an active attack, and cost protection for scale-out incurred during a mitigated attack. DDoS IP Protection offers the same mitigation on a per-public-IP basis for smaller estates.

The relevant scoping question is simple: DDoS protection applies to public IP addresses. In an estate where compute has no public addressing and ingress is consolidated behind Front Door and a small number of gateways, the protected surface — and therefore the cost — is small. This is another argument for consolidating ingress.

5.8 Network-plane telemetry

Controls that cannot be observed cannot be operated. At minimum:

- Virtual network flow logs enabled at VNet scope across the estate, with Traffic Analytics for enrichment and for identifying flows to malicious or unexpected public addresses.
- Azure Firewall structured logs — network rule, application rule, DNAT, DNS proxy and IDPS categories — retained in Log Analytics for the period your framework requires.
- Diagnostic settings enforced by `deployIfNotExists` policy on every network resource type. Manual enablement guarantees blind spots as the estate grows.
- Alerting on: rule changes permitting Any/Internet sources, new public IP associations, private endpoint approvals to external tenants, SNAT port utilisation thresholds, and IDPS deny events by signature.

6. Part II — Compute and Workload Hardening

6.1 Platform integrity for virtual machines

Boot-level integrity is the foundation everything above it assumes. Azure provides it through Trusted Launch, which combines secure boot, a virtual TPM and measured boot on Generation 2 virtual machines.

- **Secure boot** validates signatures on boot components, blocking unsigned bootloaders and rootkits that persist below the operating system.
- **vTPM** provides a dedicated, isolated measurement store, enabling remote attestation of boot state and supporting measured-boot integrity monitoring in Microsoft Defender for Cloud.
- **Boot integrity monitoring** compares attested measurements against expected values and alerts on divergence — the control that turns attestation from a capability into a detection.

Trusted Launch should be the default for all new virtual machines and scale sets, enforced by policy. It is free, and its principal constraint is the Generation 2 requirement, which also gates several other modern capabilities.

Confidential computing

Where the threat model includes the hypervisor and the cloud operator — a live concern in some government and regulated contexts — confidential virtual machines built on AMD SEV-SNP encrypt memory with keys held in hardware and inaccessible to the host. Confidential disk encryption extends this to the OS disk, with the key protected by the virtual TPM and released only to an attested guest. The cost is a constrained SKU and region selection and some operational complexity; the benefit is a genuine reduction in the trusted computing base.

6.2 Image supply chain

An immutable, versioned image pipeline is the difference between a hardened fleet and a fleet that was hardened once. The reference pipeline:

19. Start from a Microsoft-published marketplace image, pinned by exact version rather than by "latest".
20. Apply a hardening baseline — CIS Benchmark or, for federal work, the applicable DISA STIG — through an automated build (Azure Image Builder or Packer) whose configuration lives in source control.
21. Scan the built image for vulnerabilities and configuration drift, and fail the build on findings above threshold.
22. Publish to Azure Compute Gallery as a new immutable image version, replicated to every region the fleet spans, and mark superseded versions as excluded from latest.
23. Enforce with policy: deny virtual machine creation from any source other than the approved gallery. Without this step the pipeline is optional, and optional pipelines are bypassed under deadline.

Rebuild on a fixed cadence — monthly is a reasonable default — and out of cycle for critical vulnerabilities. Roll the fleet by replacement, not by in-place patching, so that the running state and the declared state stay convergent. For stateless workloads, ephemeral OS disks reinforce this by making local persistence impossible.

6.3 Encryption at rest

Azure offers several encryption mechanisms that are frequently conflated. They protect different things and compose rather than compete.

Mechanism	What it protects	Key custody	When to use
Storage Service Encryption with platform-managed keys	Managed disk data at rest in the storage infrastructure.	Microsoft	Always on by default; no action required.
SSE with customer-managed keys (disk encryption set)	Same scope, with the key under customer control and revocable.	Customer, in Key Vault or Managed HSM	Where a control framework requires customer key custody or cryptographic erase capability.
Encryption at host	Temp disk, OS and data disk caches — data that never reaches the storage service and is otherwise unencrypted.	Follows the disk encryption set	Always, where the SKU supports it. This closes a real and commonly overlooked gap.
Azure Disk Encryption (BitLocker / dm-crypt)	In-guest volume encryption.	Customer, in Key Vault	Legacy requirement or explicit in-guest encryption mandate. Generally superseded by host-based encryption.
Confidential disk encryption	OS disk, with the key bound to an attested confidential guest.	Customer, released on attestation	Confidential VM workloads where the host is untrusted.

For customer-managed keys, configure automatic key rotation on the disk encryption set, and understand the operational consequence of revocation: revoking a key renders the disks unreadable, which is the intended behaviour and also a self-inflicted outage if exercised without process.

6.4 Administrative access paths

Exposed management ports remain, year after year, among the most common initial access vectors in cloud environments. The controls are well understood and rarely fully applied.

- **No public IP addresses on compute.** Enforce with a deny policy at management group scope. Exceptions should be time-bounded policy exemptions with named owners, not permanent design decisions.
- **Azure Bastion for interactive access.** Bastion terminates RDP and SSH over TLS in the browser or through the native client, with no inbound port exposure on the target. Higher tiers add native client tunnelling, IP-based connection, private-only deployment and session recording — the last being the control that turns interactive access from an unauditable gap into evidence.

- **Just-in-time VM access.** Defender for Cloud's JIT capability keeps management ports denied by default and opens them, from a specified source, for a bounded window on approved request. Every request is logged and attributable.
- **Microsoft Entra ID authentication to the guest.** The Entra login extension replaces local accounts with directory identities on Linux and Windows, bringing Conditional Access, phishing-resistant multi-factor authentication and centralised revocation to SSH and RDP. Removing local privileged accounts eliminates a credential class that no cloud control can see.
- **Privileged Identity Management on the control plane.** Owner and Contributor at subscription or management group scope should be eligible, not permanent — activated with justification, approval and a time limit. Standing subscription-wide privilege is the mechanism by which a single compromised administrator account becomes an estate-wide incident.

6.5 Patch and configuration management

Azure Update Manager is the current native service for assessment and deployment of operating system updates across Azure, Arc-connected and on-premises machines. It succeeded Automation Update Management and requires no dedicated Log Analytics workspace.

- Enable periodic assessment fleet-wide so that patch state is continuously known rather than discovered during a scan.
- Define maintenance configurations per environment with distinct windows, and use customer-managed schedules so that patching is deterministic rather than dependent on per-VM settings.
- Treat patching as a compensating control for the image pipeline, not a replacement for it. Long-lived machines accumulate drift regardless of patch currency.

For in-guest configuration, machine configuration (the successor to guest configuration) audits and remediates operating system settings through Azure Policy, giving a single compliance surface across control-plane and in-guest state. Combine with **Microsoft Defender for Servers** for vulnerability assessment, endpoint detection and response through Defender for Endpoint, file integrity monitoring and agentless scanning — capabilities differ between the P1 and P2 plans, and the JIT, file integrity and adaptive controls that matter most for hardening sit in P2.

6.6 Azure Kubernetes Service

AKS concentrates more security-relevant configuration in one resource than anything else in Azure. A default cluster is not a hardened cluster, and several of the defaults are actively dangerous in a production context.

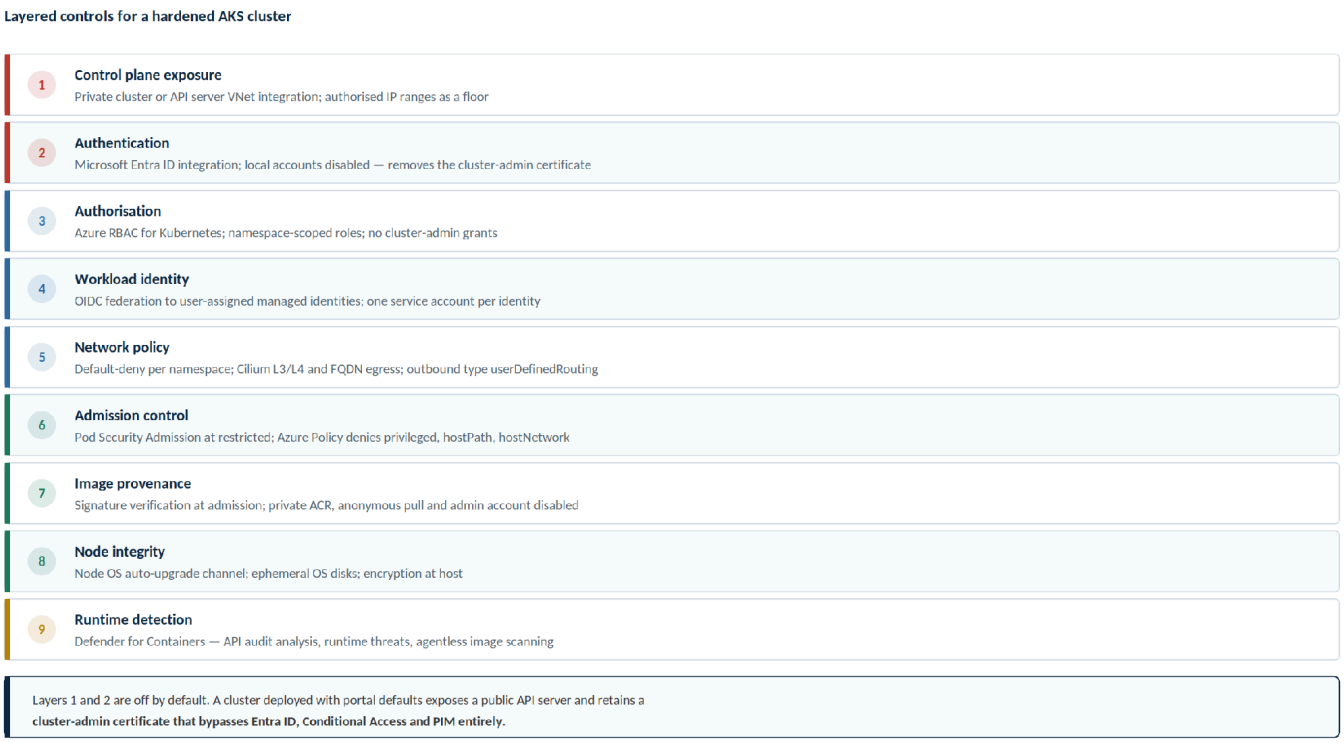

Figure 4 — Layered controls for a hardened AKS cluster. The layers are independent: weakening any one of them provides a path that the others were not designed to catch.

Control plane exposure

- **Private cluster or API server VNet integration.** A public API server endpoint, even with authentication, is a discoverable, brute-forceable, and version-fingerprintable surface. Private clusters expose the API server through Private Link; API server VNet integration projects it into a delegated subnet without the private DNS complexity. Where neither is viable, authorised IP ranges are the minimum acceptable control.
- **Disable local accounts.** By default AKS retains a cluster-admin client certificate that authenticates outside Microsoft Entra ID entirely — no Conditional Access, no multi-factor authentication, no PIM, and no practical revocation. Deploying with local accounts disabled removes it. This is the single most important AKS security setting and it is off by default.
- **Entra ID integration with Azure RBAC for Kubernetes authorisation.** This unifies cluster authorisation with the Azure control plane, so that access reviews, PIM eligibility and audit trails cover the cluster as well as the subscription. Avoid granting the cluster-admin role; scope to namespaces with the reader, writer and admin built-ins.

Workload identity

Microsoft Entra Workload ID is the current mechanism for pods to authenticate to Azure services. The cluster acts as an OIDC issuer; a projected service account token is exchanged for an Entra token against a federated credential on a user-assigned managed identity. No secret exists in the cluster at any point. Its predecessor,

pod-managed identity, was deprecated in 2022 and the managed add-on was supported only through September 2025 — any cluster still using it is unsupported.

- Map one Kubernetes service account to one managed identity. Sharing an identity across service accounts simplifies role assignment and destroys blast-radius isolation, which is the entire point.
- Note the platform limit on federated identity credentials per managed identity — around twenty — and plan the service-to-identity mapping before a large microservice estate hits it.
- Migrating does not remove existing Kubernetes secret objects. Run both paths in parallel, verify, then explicitly delete the old secrets and the environment variables and volume mounts that consumed them.

Networking

- **Azure CNI Overlay.** kubenet is scheduled for retirement on 31 March 2028; new clusters should use CNI Overlay, which conserves VNet address space while preserving routable pod networking semantics.
- **Network policy enforced, default-deny per namespace.** Without network policy, any pod can reach any other pod in the cluster. Azure CNI powered by Cilium provides efficient L3/L4 enforcement and, in supported configurations, FQDN-based egress policy.
- **Egress through the firewall.** Set the cluster outbound type to user-defined routing and route egress to the hub firewall, allow-listing the AKS-required FQDNs Microsoft publishes. Note that from 31 March 2026 clusters using the AKS-managed VNet option place cluster subnets into private subnets by default.

Node and admission hardening

- Enable node OS auto-upgrade on a security-patch or node-image channel so that nodes do not silently age out of support. Combine with ephemeral OS disks and host-based encryption.
- Enforce Pod Security Admission at the `restricted` level for application namespaces, and layer Azure Policy for AKS (Gatekeeper-based admission control) to deny privileged containers, host networking, host path mounts and privilege escalation, and to require read-only root filesystems and dropped capabilities.
- **Verify image provenance at admission.** Signature verification through a Ratify/Notation-based policy ensures that only images signed by your build system can run. Pair with a private Azure Container Registry: private endpoint, anonymous pull disabled, admin account disabled, and repository-scoped tokens rather than shared credentials.
- **Secrets from Key Vault, not from etcd.** The Secrets Store CSI driver mounts Key Vault secrets into pods at runtime. Where secrets must exist as Kubernetes objects, enable KMS etcd encryption with a customer-managed key.
- **Defender for Containers.** Provides runtime threat detection, Kubernetes API audit analysis and agentless vulnerability assessment of registry images. Runtime detection is the layer that catches what admission control did not anticipate.

6.7 App Service and Azure Functions

App Service is deceptively easy to deploy insecurely, because its defaults optimise for a developer publishing a site rather than an enterprise running a regulated workload.

- **Disable the public data plane and use a private endpoint.** Set public network access to disabled; a private endpoint alone does not close the public path.
- **Route all outbound traffic through the virtual network.** Regional VNet integration by default routes only private address ranges through the VNet. Enabling the route-all setting is what actually forces application egress through your firewall — without it, the application reaches the internet directly and the egress design does not apply to it.
- **Remember the SCM site.** The Kudu/SCM endpoint is a separate hostname with its own access restrictions, and it offers a console, a file browser and deployment capability. Restricting the application site while leaving SCM open is a frequent and serious oversight.
- **Disable basic authentication for SCM and FTP.** These credential paths bypass Entra ID entirely. Disable them explicitly — there is a dedicated resource type for it — and enforce with policy. Disable FTP, and disable remote debugging in all non-development slots.
- **Eliminate secrets from application settings.** Use a system-assigned managed identity with Key Vault references so that the setting holds a reference, not a value, and the value is resolved at runtime under an auditable identity.
- **Enforce transport security.** HTTPS-only, a minimum TLS version of 1.2 or higher, and client certificate mode set where mutual TLS is appropriate.
- **For Functions, treat the backing storage account as part of the trust boundary.** The function host stores code, keys and state there. Use identity-based connections rather than a connection string, restrict the account with a private endpoint, and disable its public access.

6.8 Secrets and key management

The strategic objective is not better secret storage. It is fewer secrets. Managed identities for Azure-to-Azure authentication, and federated credentials for CI/CD systems such as GitHub Actions and Azure DevOps, remove entire classes of long-lived credential. What remains — third-party API keys, certificates, legacy system credentials — belongs in Key Vault.

The access control model is changing

Key Vault has two authorisation models: legacy access policies, and Azure RBAC. Azure RBAC is the recommended model and is now the default for newly created vaults from control plane API version `2026-02-01` onward. All Key Vault control plane API versions prior to `2026-02-01` retire on **27 February 2027**. Existing vaults keep their current model unless explicitly changed, and both models remain supported — but there are two concrete actions to take now:

24. Audit infrastructure-as-code and automation that creates vaults. Under the newer API, a template that omits `enableRbacAuthorization` will produce an RBAC vault; if the accompanying role assignments are absent, deployments and applications fail with authorisation errors that look like outages rather than configuration changes.
25. Update control plane management SDKs and pinned API versions ahead of the February 2027 retirement. Data plane SDKs are unaffected.

Migrating an existing vault to RBAC invalidates all access policy permissions at the moment of the switch. Enumerate every principal and its effective permissions, map them to equivalent built-in roles, stage the assignments, then flip the model.

Vault configuration baseline

- **Purge protection enabled.** Soft delete is always on; purge protection prevents a deleted vault or key from being permanently destroyed within the retention window. It is irreversible once enabled, which is precisely why it is the control — it removes destructive deletion as an attack outcome.
- **Public network access disabled, private endpoint deployed.** Apply the trusted-services bypass deliberately rather than by default; it is a broad exception.
- **One vault per trust and blast-radius boundary.** Per-application, per-environment vaults, not one organisational vault. RBAC can scope to individual secrets, but a single vault remains a single availability and compromise domain.
- **Rotation policies with Event Grid notification.** Rotation that depends on a calendar reminder does not happen.
- **Managed HSM where key custody requirements are stringent.** Managed HSM provides single-tenant, FIPS 140-3 Level 3 validated hardware with a customer-controlled security domain. It is the right answer for high-assurance and many federal workloads, and overkill for most others.

7. Part III — Enforcement: Policy as the Control Plane

Everything in Parts I and II is a configuration state. Configuration states drift. Azure Policy is the mechanism that makes a control continuous rather than instantaneous, and it is the difference between a hardening project and a hardened platform.

7.1 Effects and where each belongs

Effect	Behaviour	Use for
Audit / AuditIfNotExists	Records non-compliance; changes nothing.	Discovery, baselining before enforcement, and controls where blocking is disproportionate.
Deny	Rejects the resource creation or update request.	Non-negotiable controls: public IPs on compute, permissive NSG rules, unapproved regions, non-gallery images.
DeployIfNotExists	Deploys a remediating resource when the condition is unmet.	Diagnostic settings, private DNS zone registration, agent and extension deployment, flow log enablement.
Modify	Alters properties on create or update, and can remediate existing resources.	Enforcing tags, TLS minimums, and other property-level defaults.
DenyAction	Blocks a specified action, most usefully deletion.	Protecting log destinations, key vaults and other resources whose deletion is itself an attack.

7.2 A rollout pattern that does not cause outages

26. Deploy the definition in audit mode at management group scope and leave it for a full change cycle.
27. Quantify non-compliance. If a deny would have blocked a material share of legitimate deployments, the definition is wrong, not the estate.
28. Issue exemptions for genuine exceptions — **always with an expiry date and a named owner**. An exemption without an expiry is a permanent silent hole in the control.
29. Switch to deny. Communicate ahead of the change with the specific error message teams will encounter and the remediation path.
30. Review exemptions on a fixed cadence. Expiring exemptions are the mechanism that keeps the control set honest over time.

7.3 Initiatives and frameworks

Do not author from scratch what Microsoft maintains. The Microsoft Cloud Security Benchmark is assigned by default and is a reasonable floor. Built-in regulatory compliance initiatives exist for NIST SP 800-53 Rev. 5, NIST SP 800-171, CMMC, FedRAMP High and the DoD Impact Levels, and they double as evidence generators —

the Defender for Cloud regulatory compliance dashboard maps control status to framework controls in a form assessors accept.

Layer a custom initiative on top for organisation-specific controls: approved regions, approved image sources, mandatory tags, private endpoint requirements. Keep the custom set small; each definition is something you now maintain.

7.4 What policy cannot do

Azure Policy evaluates resource properties at the Resource Manager layer. It does not see inside application code, cannot evaluate most in-guest state without machine configuration, and cannot reason about intent. It must be paired with:

- Pre-deployment static analysis of infrastructure-as-code — Checkov, tfsec, PSRule for Azure — so that violations surface in a pull request rather than as a deployment failure.
- Deployment stacks with deny settings, which prevent out-of-band modification and deletion of managed resources and give a coherent lifecycle boundary. Resource locks are a blunt operational safeguard, not a security control — they are trivially removed by anyone holding the permission to remove them.
- Peer review on the definitions themselves. A policy set is code with production authority and deserves the same review rigour as the workloads it governs.

8. Part IV — Azure Government Deltas

Azure Government is a physically and logically separate cloud instance, not a region of Azure Commercial. Designs, tooling and automation port across with effort, and the failure modes are consistent enough to enumerate.

8.1 It is a different cloud, not a different region

Separate identity, separate portal, separate management and data plane endpoints. Anything that hardcodes a Commercial endpoint will fail, and some failures are quiet.

Surface	Azure Commercial	Azure Government
Portal	portal.azure.com	portal.azure.us
Entra ID authority	login.microsoftonline.com	login.microsoftonline.us
Resource Manager	management.azure.com	management.usgovcloudapi.net
Microsoft Graph	graph.microsoft.com	graph.microsoft.us
Key Vault data plane	vault.azure.net	vault.usgovcloudapi.net
Blob storage	blob.core.windows.net	blob.core.usgovcloudapi.net

Tooling must be told which cloud it is addressing: `az cloud set --name AzureUSGovernment` for the CLI, the `environment = "usgovernment"` provider argument in Terraform, and the equivalent environment setting in each SDK. Build one codebase parameterised by cloud rather than two divergent ones.

> **GOV DELTA — PRIVATE DNS ZONE NAMES ARE NOT A FIND-AND-REPLACE**
>
> The most common Azure Government private endpoint failure is assuming a uniform suffix substitution. It is not uniform. Storage private endpoints use privatelink.blob.core.usgovcloudapi.net, while monitoring uses privatelink.monitor.azure.us and several other services use .azure.us or service-specific .us domains. A blanket rewrite of windows.net to usgovcloudapi.net produces zones that look plausible, resolve to nothing, and fail intermittently depending on which service a workload touches. Derive zone names from the current Microsoft reference table per service, and validate resolution from inside the VNet before declaring the endpoint working.

8.2 Compliance posture

The compliance framing is the reason most organisations are in Azure Government, and it should drive region selection explicitly.

- **FedRAMP High.** Azure Government holds a FedRAMP High provisional authorisation to operate issued by the Joint Authorization Board, covering the US Gov Arizona, Texas and Virginia regions.

- **DoD Impact Levels.** Provisional authorisations exist at IL2, IL4 and IL5. Separate IL5 authorisations apply to the US Gov regions and to the US DoD Central and US DoD East regions, the latter reserved for exclusive Department of Defense use. IL6, covering classified information up to SECRET, is served by Azure Government Secret rather than Azure Government.
- **Personnel restrictions.** The DoD Cloud Computing SRG restricts cloud provider personnel with access to IL4 and IL5 data to US citizens, US nationals or US persons. This is often the determining requirement rather than any technical control.
- **IL5 isolation.** Some services in the US Gov regions require additional configuration to meet IL5 compute and storage isolation requirements — commonly customer-managed keys in Key Vault for encryption at rest, and dedicated or isolated compute. Microsoft publishes per-service isolation guidance; treat it as a design input, not a post-deployment checklist.

8.3 Service availability and feature lag

Not every Azure service exists in Azure Government, and among those that do, feature parity often lags Commercial by one or more release cycles. Preview features are frequently Commercial-only. This affects hardening designs directly: a control that depends on a recently released capability may not be available, and the compensating control needs to be designed in from the start rather than discovered at deployment.

Practical guidance: verify current availability against the products-by-region reference at design time for every service in the architecture, and re-verify before each major release. Build the architecture so that Commercial-only capabilities are additive enhancements rather than load-bearing assumptions.

8.4 Operational differences that surface late

- Marketplace and image availability differ. An image pipeline built against Commercial marketplace images will not port directly; build the Gov pipeline against Gov-available base images from the outset.
- Regulatory compliance initiatives specific to Azure Government exist for the DoD Impact Levels; assign the Gov-specific initiative rather than the Commercial equivalent.
- ExpressRoute peering locations and circuit types differ, and Gov circuits have their own procurement path and lead times.
- Support escalation paths, engineering access and diagnostic data handling are subject to screened-personnel requirements, which lengthens some support interactions.
- Cross-cloud connectivity between Commercial and Government requires explicit design; it is not implicit and should be treated as an external network boundary with corresponding controls.

9. Implementation Roadmap

Attempting every control simultaneously produces a stalled programme. The sequencing below front-loads the controls that close the highest-probability attack paths and defers those that require the most engineering investment. Timeframes assume a mid-sized enterprise estate with a dedicated platform team.

Phase	Objective	Principal controls	Signal of completion
0 — Visibility (0–30 days)	Know the actual state before changing it.	Defender for Cloud enabled across subscriptions; VNet flow logs and diagnostic settings deployed by policy; inventory of public IPs, publicly accessible PaaS data planes, permissive NSG rules, unmanaged VMs, vaults on access policies, and AKS clusters with local accounts enabled.	A defensible, current inventory of exposure that the security and platform teams both accept.
1 — Close the front door (30–90 days)	Remove direct inbound access to compute and management surfaces.	Deny policy on public IPs for compute; Bastion and JIT for interactive access; basic authentication and FTP disabled on App Service; TLS minimums enforced; WAF deployed in detection mode; PIM on subscription-scope roles.	No compute instance in the estate is directly reachable from the internet; all interactive access is brokered and logged.
2 — Control egress and the data plane (90–180 days)	Every outbound path is inspected; PaaS data planes leave the internet.	Azure Firewall with UDR forced tunnelling; FQDN allow-listing and IDPS in alert mode; private endpoints with public network access disabled; centralised private DNS with deployIfNotExists registration; WAF to prevention mode.	Egress is enumerable and auditable; no PaaS resource in scope accepts public data plane traffic.
3 — Workload integrity (180–365 days)	Compute is built, attested and credentialed to a standard.	Versioned image pipeline with a hardening baseline; Trusted Launch enforced; encryption at host; workload identity migration completed; AKS admission control and network policy; customer-managed keys where required.	No workload authenticates with a long-lived secret; no instance runs an image outside the pipeline.
4 — Enforce and evidence (continuous)	Controls hold without ongoing manual attention.	Audit-mode policies converted to deny; exemption review cadence established; IaC static analysis gating pull requests; deployment stacks with deny settings; regulatory compliance dashboard as the reporting surface.	Drift is prevented rather than detected; compliance evidence is generated by the platform rather than assembled by people.

9.1 A note on sequencing

Phase 2 is where most programmes stall, because egress control is the first phase whose failures are visible to application teams. The mitigation is procedural rather than technical: build the FQDN allow-list collaboratively

with application owners during phase 1, run the firewall in a permissive logging posture for a full cycle before enforcing, and publish the request path for adding a destination before the first denial occurs. Egress control fails on organisational readiness far more often than on engineering.

Appendix A — Hardening Checklist

A condensed control list for design review and assessment. Each item maps to the section that discusses its rationale and trade-offs.

Domain	Control	§
Network	Explicit intra-VNet deny beneath scoped allows; no Any/Any rules anywhere in the estate.	5.2
Network	Application security groups used in place of IP literals; regional service tags preferred over broad ones.	5.2
Network	Virtual network flow logs enabled at VNet scope; NSG flow logs migrated ahead of retirement.	5.2
Network	Explicit outbound method on every subnet; default route to an inspection point.	5.3
Network	Azure Firewall with FQDN application rules; network rules constrained so they do not shadow them.	5.4
Network	IDPS enabled, tuned in alert mode, then moved to alert-and-deny.	5.4
Network	Private endpoints deployed AND public network access disabled on every supported PaaS resource.	5.5
Network	Private DNS zones hosted centrally with policy-driven A-record registration; on-premises conditional forwarders configured.	5.5
Network	Private endpoint creation restricted by policy to approved resource types and target subscriptions.	5.5
Network	WAF in prevention mode with narrowly scoped exclusions; origin restricted so the WAF cannot be bypassed.	5.6
Compute	Trusted Launch enforced on all Generation 2 virtual machines; boot integrity monitoring enabled.	6.1
Compute	All instances built from a versioned Compute Gallery image; non-gallery sources denied by policy.	6.2
Compute	Encryption at host enabled; customer-managed keys with automatic rotation where required.	6.3
Compute	No public IP addresses on compute; Bastion and just-in-time access for all interactive sessions.	6.4
Compute	Entra ID guest authentication; local privileged accounts removed; PIM on subscription-scope roles.	6.4
Compute	Periodic patch assessment enabled fleet-wide with defined maintenance configurations.	6.5
AKS	Private cluster or API server VNet integration; local accounts disabled; Entra ID with Azure RBAC authorisation.	6.6

Domain	Control	§
AKS	Workload identity in use; pod identity fully removed; one service account per managed identity.	6.6
AKS	Default-deny network policy per namespace; egress via user-defined routing to the firewall.	6.6
AKS	Pod Security Admission at restricted level plus Azure Policy admission control; image signature verification enforced.	6.6
PaaS	App Service public access disabled with private endpoint; route-all outbound enabled; SCM endpoint restricted.	6.7
PaaS	Basic authentication and FTP disabled; remote debugging off; HTTPS-only with a minimum TLS version enforced.	6.7
Secrets	Key Vault on Azure RBAC; control plane API versions updated ahead of the February 2027 retirement.	6.8
Secrets	Purge protection enabled; public network access disabled; one vault per trust boundary.	6.8
Secrets	Managed identities and federated credentials replacing long-lived secrets across applications and pipelines.	6.8
Enforcement	Deny policies at management group scope for non-negotiable controls; all exemptions carry an expiry and an owner.	7.2
Enforcement	Regulatory compliance initiative assigned and used as the evidence surface.	7.3
Enforcement	IaC static analysis gating pull requests; deployment stacks with deny settings on managed resources.	7.4
Government	Endpoints and private DNS zone names derived per service; no blanket suffix substitution.	8.1
Government	Service availability verified per service at design time; IL5 isolation guidance applied where in scope.	8.2

About Cettabyte

Cettabyte is a consulting practice specialising in cloud, software, and AI. We work with enterprise, mid-market and public sector organisations to build, modernize, and secure across the whole stack.

This paper reflects patterns we encounter repeatedly in assessment and remediation work.

cettabyte.com

Azure service capabilities, defaults and lifecycle dates change frequently. Retirement and default-behaviour dates cited in this paper were current as of publication in July 2026 and should be re-verified against current Microsoft documentation before being relied upon in a design decision.