CHECKLIST

AWS Virtualization Security Best Practices for EC2 Isolation Checklist

A practical checklist for validating EC2 isolation across tenancy, network segmentation, IAM boundaries, logging, and recovery assumptions before production.

AWS Virtualization Security Best Practices for EC2 Isolation

Production Readiness Checklist

Use this checklist to validate whether an EC2 workload is isolated enough for its sensitivity, operational impact, and recovery requirements.

---

1) Classify the workload

- [ ] Identify the data types the instance will process or store.
- [ ] Mark whether the workload handles regulated, confidential, or high-value data.
- [ ] Define the blast radius if the instance is compromised.
- [ ] Determine whether compromise could affect adjacent systems, credentials, or production pipelines.

- [ ] Record whether the workload is stateless, stateful, or control-plane adjacent.

Pass criteria: You can explain why the workload belongs on EC2 and what would be harmed by a compromise.

---

2) Verify the compute isolation model

- [ ] Confirm the instance family and platform capabilities used in the environment.
- [ ] Verify the tenancy model: shared, dedicated, or host-based placement as applicable.
- [ ] Confirm whether the workload requires stronger isolation than the default shared-host model.
- [ ] Validate region- and instance-family-specific behavior before treating hardware-backed isolation as a control.
- [ ] Document what isolation guarantees are assumed versus explicitly verified.

Pass criteria: The tenancy and compute placement model are known, intentional, and aligned with the workload risk.

---

3) Review network exposure and segmentation

- [ ] Map all inbound paths to the instance.
- [ ] Map all outbound paths from the instance.
- [ ] Restrict inbound access to only required sources and ports.
- [ ] Confirm the instance is in the correct subnet and routing domain.
- [ ] Separate production, non-production, and administrative paths.
- [ ] Verify there are no unnecessary management ports exposed to broad networks.
- [ ] Check whether security groups, NACLs, and routes collectively enforce the intended boundary.

Pass criteria: Every network path to and from the instance is intentional and minimized.

---

4) Scope identity and credential boundaries

- [ ] Review the IAM role attached to the instance.
- [ ] Remove broad permissions that are not essential to the workload.
- [ ] Confirm the instance cannot read secrets it does not need.
- [ ] Verify the role cannot modify security boundaries unless explicitly required.
- [ ] Check whether any long-lived credentials exist inside the guest.
- [ ] Eliminate shared administrative credentials where possible.
- [ ] Confirm credential rotation and revocation processes are documented.

Pass criteria: A compromised guest would have limited authority and limited access to sensitive resources.

---

5) Reduce guest-level pivot risk

- [ ] Verify OS access is restricted to approved administrators.
- [ ] Remove unnecessary sudo or root access paths.
- [ ] Harden remote access and disable unused services.
- [ ] Confirm the instance cannot directly act as a trusted control-plane component.
- [ ] Validate that agents, automation tools, and scheduled tasks have minimal permissions.
- [ ] Review whether the guest can reach build systems, deployment systems, or identity systems.

Pass criteria: Guest compromise does not provide an easy pivot into administration or production control.

---

6) Validate logging and detection

- [ ] Enable logging for administrative and security-relevant activity.
- [ ] Confirm guest activity can be correlated with network and identity events.
- [ ] Alert on privilege escalation, unexpected credential access, and unusual outbound connections.
- [ ] Monitor changes to security groups, instance profiles, and snapshots.
- [ ] Verify logs are retained in a separate, protected location.
- [ ] Test that alerts reach the correct operational responders.

Pass criteria: Compromise indicators are observable and actionable.

---

7) Test recovery and containment assumptions

- [ ] Confirm the instance can be rebuilt from a clean source.
- [ ] Verify replacement is easier than recovery-in-place.
- [ ] Document how to revoke credentials quickly after compromise.
- [ ] Test the ability to isolate or terminate the instance without delaying incident response.
- [ ] Validate snapshot, AMI, and image handling procedures.
- [ ] Ensure old artifacts do not preserve unintended trust.
- [ ] Confirm backups and images are stored and shared according to the same sensitivity rules as production data.

Pass criteria: A suspected compromise can be contained and recovered from quickly.

---

8) Check for hidden persistence channels

- [ ] Review whether snapshots are retained longer than necessary.
- [ ] Review whether AMIs or images are copied across accounts without clear ownership.
- [ ] Confirm temporary build or test instances do not leave trusted artifacts behind.
- [ ] Remove unused keys, tokens, and automation credentials from images.
- [ ] Validate that image pipelines do not reintroduce broad permissions.

Pass criteria: Old artifacts do not weaken the isolation boundary after the instance is gone.

---

9) Decide whether stronger isolation is required

Use stronger controls when any of the following are true:

- [ ] The workload processes regulated, confidential, or high-value data.
- [ ] A compromise could reach adjacent systems, secrets, or production pipelines.
- [ ] The instance hosts privileged tooling, agents, or administrative access.
- [ ] The environment requires strict tenancy or separation of duties.
- [ ] Recovery must avoid cross-workload contamination.

If one or more apply:

- [ ] Reassess tenancy and placement options.
- [ ] Compare available hardware-backed or dedicated isolation features.
- [ ] Confirm the exact behavior of those features in the target region and instance family.
- [ ] Document the control as a verified requirement, not an assumption.

Pass criteria: The isolation model matches the workload?s actual risk.

---

10) Production admission check

Before approving the workload for production, confirm the following:

- [ ] The blast radius is documented.
- [ ] The tenancy model is intentional.
- [ ] Network exposure is minimized.
- [ ] IAM permissions are least privilege.
- [ ] OS-level access is controlled.
- [ ] Logging and detection are enabled.
- [ ] Recovery can happen quickly.
- [ ] Snapshots and images are governed.
- [ ] The workload has a named owner for security decisions.

Ready for production when: You can describe the workload boundary in one sentence and prove the controls that enforce it.

Quick validation summary

Area	What to verify	Status
Workload sensitivity	Data type and blast radius documented	?
Compute isolation	Tenancy and placement confirmed	?
Network segmentation	Ingress and egress minimized	?
Identity boundaries	IAM and credentials scoped tightly	?
Guest hardening	Admin paths and pivots restricted	?
Detection	Logs and alerts in place	?
Recovery	Rebuild and revocation tested	?
Artifact hygiene	Snapshots and AMIs governed	?

Final rule of thumb

EC2 isolation is strong enough for many workloads when the instance type, placement model, network paths, and identity scope are all explicit. If you cannot clearly explain one of those layers, the isolation boundary is probably too implicit to trust.