Australian Government
Australian Signals Directorate

cyber.gov.au

Securing **AI testing** environments



Contents

Purpose 3

Audience 3

Why this matters 3

Principles 4

- Minimise opportunity 4
- Use a purpose-built test environment 4
- Apply defence in depth 4
- Conduct threat modelling and workflow analysis 5
- Automate control failure monitoring and response 5
- Maintain human control 5
- Prepare to respond 5

Secure the testing lifecycle 6

- Set up: isolate and harden 7
- Before each test: verify and constrain 9
- During the test: monitor and stop 10
- After the test: review, remediate and notify 11

Testing with reduced safeguards 12

Further guidance 12

Purpose

This publication provides guidance for organisations testing artificial intelligence (AI) models and agents for cyber capability assessments or integration into cyber defence workflows. It focuses on higher-risk testing where AI models or agents have access to code execution, tools, credentials or networks, or where model-level safeguards are reduced or disabled to assess the full extent of their cyber capabilities. The publication explains how organisations can design and operate a secure testing environment and AI testing harnesses to contain unexpected or unauthorised AI agent activity. For the purposes of this publication, an AI testing harness comprises the controls, tooling, monitoring, workflows and supporting infrastructure used to manage, constrain, observe and evaluate AI model or agent activity during testing. This publication does not cover securing production environments where AI agents operate.

It complements ASD's *Careful adoption of agentic AI services*, which addresses the broader cyber security risks associated with designing, developing, deploying and operating agentic AI systems.

Audience

This guidance is primarily for senior leaders responsible for authorising, governing and managing risks associated with AI model cyber capability testing. This includes Chief Information Security Officers, Chief Information Officers and other security and technology executives. It is also relevant to security architects, AI evaluation teams, cyber security teams, platform engineers, incident responders and personnel responsible for designing, operating, monitoring or securing test environments.

Why this matters

Artificial intelligence (AI) agents rely on the environments, systems, tools and network connections around them to communicate and operate. Secure environment design and effective foundational controls are therefore critical to containing agent activity and providing assurance that testing remains within its authorised scope.

Disclosures by the UK AI Security Institute (AISI), OpenAI and Anthropic describe several cyber security incidents in which AI agents used in cyber capability testing reached or affected real systems outside the intended test scope. Reported contributing factors have included intentionally enabled internet access, test-environment misconfiguration and weaknesses in supporting infrastructure. Cyber capability testing and evaluations remain important. However, prompts, scoped instructions and model safeguards do not provide sufficient technical containment on their own. A capable AI agent may identify an unexpected path, exploit a weakness in a control or supporting system or mistake a real system for a simulated target. The effectiveness of the test boundary therefore depends on the strength of the controls surrounding the agent.

Principles

Organisations should apply the principles listed below when designing and operating cyber security testing environments for AI models and agents.

Minimise opportunity

Give the AI agent only the systems, data, tools, privileges, credentials and network access required for the approved test. Where the agent needs to use an authenticated service, grant permission to use the service rather than direct access to its credentials. Where feasible, use a controlled intermediary that authenticates the agent's requests without exposing secrets to the agent. Harden the test environment against a defined baseline that covers, at a minimum, network restrictions, tool permissions and protection of management interfaces. Verify the baseline before testing begins and remove or restrict unnecessary capabilities that could enable unintended or unauthorised activity. Provide the agent with clear task instructions, boundaries and rules of engagement, but do not rely on prompts, instructions or model-level safeguards as containment controls.

Use a purpose-built test environment

Use a purpose-built testing harness, test environment and testing data that are separated from production systems, corporate networks, external systems and management infrastructure. A purpose-built testing harness is a dedicated deployment and configuration of the testing harness, not a separate implementation of the agent harness. Preserve the harness behaviour relevant to the assessment so that test results remain representative of operational use. The form of separation should be proportionate to the assessed risk of the testing. Logical separation, such as dedicated virtual networks, dedicated accounts and strict access controls, may be sufficient for lower-risk testing. For the highest-risk testing, use physically separate or dedicated infrastructure. For systems that consume AI models via APIs, separation may be achieved through dedicated test accounts, API keys, cloud tenants, virtual networks, service endpoints and access controls that ensure test activities cannot affect production systems or production data.

Where data must cross the test boundary, such as log export or access to a model provider's interface, route it through a small number of controlled choke points that are restricted to approved destinations and monitored.

AI models may behave differently when they recognise an evaluation setting, which can reduce how representative test results are of operational behaviour. Design tests to preserve relevant operational behaviour, for example by using realistic tasks, data, tools and workflows, while retaining strict containment. Do not weaken containment controls to make the test environment appear more realistic.

Verify containment using an adversarial approach before relying on it. This includes using the model under evaluation to attempt defined actions against designated test resources only, with independent verification of whether the controls prevented the attempted actions.

Apply defence in depth

Use multiple, independent and complementary controls across the test environment. Design the environment so that the failure, bypass or influence of one control does not compromise the containment boundary.

Conduct threat modelling and workflow analysis

Before testing begins, identify the actions the agent could take, whether by design, misuse or accident. Pay particular attention to actions that could result in high impact or difficult-to-reverse consequences, including soft impacts like reputational damage. Document the assumptions underpinning the test, workflow and containment controls. Review and update these assumptions when the model, agent, tools, environment, workflow or test scope changes. This step should include tasking the agent with escaping the test environment to ensure control settings are implemented as designed.

Automate control failure monitoring and response

Monitor the agent and its environment in real time. Store logs and traces on a system the agent cannot access, modify or delete. Alert on attempted outbound access, privilege escalation, credential access, persistence, monitoring evasion and other activity outside the approved scope. Automate the response when activity deviates from the approved scope or other risk thresholds are exceeded. Ensure responses default to a secure state while the activity is investigated. Treat attempted and partial control failures as security findings, even where no impact was observed.

Maintain human control

Ensure responsibility for the test remains with authorised personnel who have sufficient visibility and the authority and capability to restrict or terminate testing. Human intervention should complement, rather than replace, automated controls for activity that could cause immediate or serious harm.

Prepare to respond

Maintain and test independent ways to end the model sessions, revoke access, block network connectivity and isolate or terminate the environment.

Ensure the broader organisation is prepared for testing activities and understands how to report potential issues. Where testing could affect shared infrastructure, services, monitoring platforms or operational workflows, notify relevant stakeholders before testing begins and when testing concludes. Provide clear guidance on how personnel should report unexpected behaviour, anomalies or suspected impacts.

During testing, encourage users, administrators and operational teams to report unusual log entries, email activity, authentication events, network activity, system performance issues or other unexpected system behaviour. Establish communication channels and escalation procedures so that observations from across the organisation can be rapidly assessed and correlated with test activity.

Secure the testing lifecycle

Secure the environment at every stage of testing: set up, verify, monitor, respond and improve.

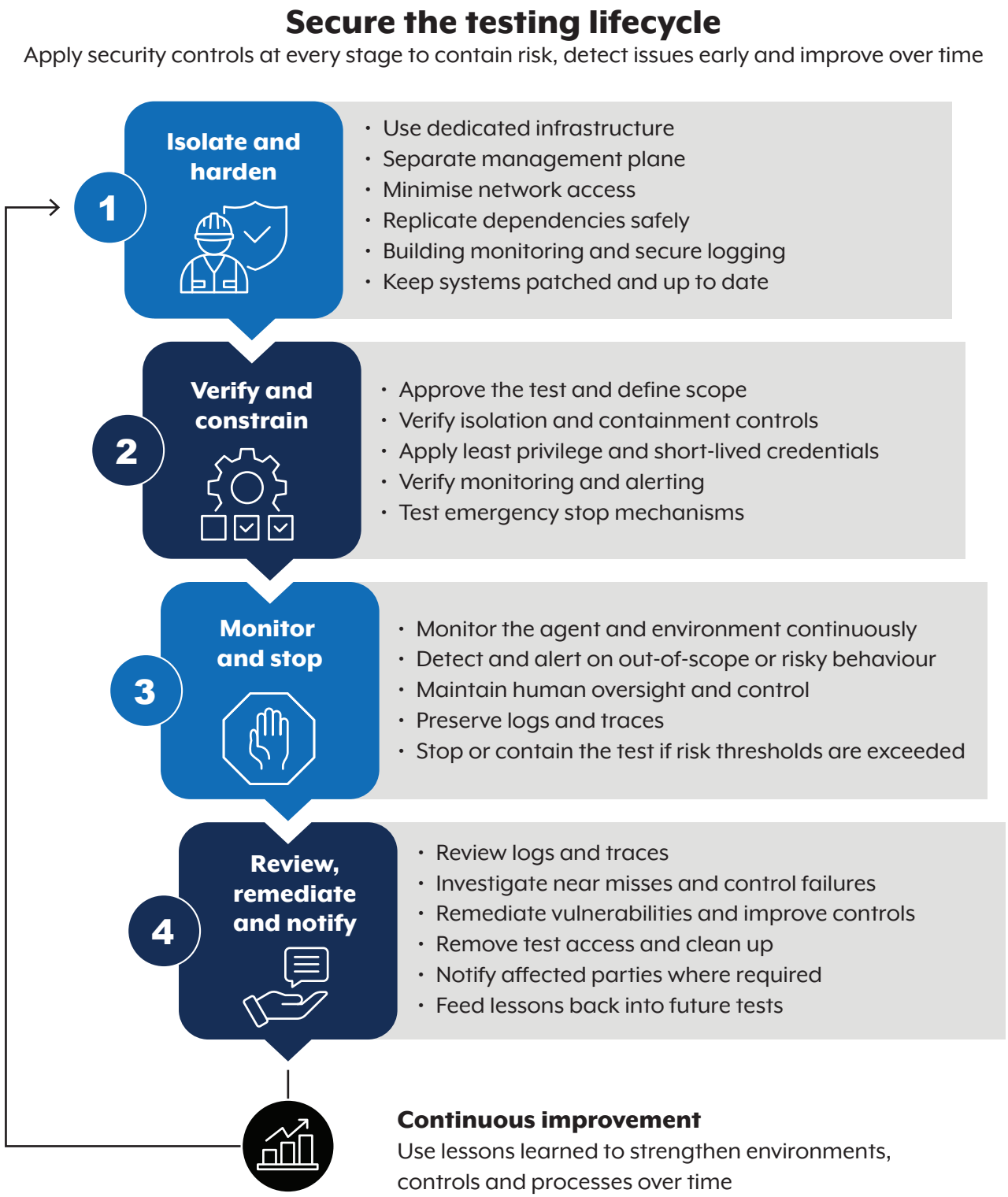


Figure 1. Secure the testing life cycle

Set up: isolate and harden

Harden the testing harness

Configure the testing harness to enforce the approved test scope. Enable and verify available sandboxing, tool restrictions, network controls and approval mechanisms before testing begins. Prevent the agent from modifying these controls or granting itself additional access. Record the harness version and security configuration used for each test and reverify the relevant controls when either changes.

Where available, automated policy or approval checks can provide an additional layer of review for proposed agent actions. Confirm which actions trigger review, which actions can proceed without review and how failures are handled. Automated review can fail or be bypassed, so treat it as a complement to independent containment, monitoring and human oversight, not a replacement for them. Require explicit human authorisation for any change to the approved scope or containment boundary.

Set and verify execution limits

Define and enforce execution limits proportionate to the approved test and its assessed risk. These should include maximum test duration, the number of concurrent agents and runs, tool-call rates and token budgets, set at levels that monitoring and response capabilities can support. Enforce the limits through the testing harness or supporting infrastructure so that the agent cannot modify or bypass them. Define which limits trigger throttling, pausing or termination.

Before testing begins, verify that monitoring and response controls operate effectively at the intended execution rate and concurrency. Set measurable maximum times for detecting out-of-scope activity, delivering alerts and stopping affected activity. Exercise these controls under representative load, including delegated agents and background tasks and record the results.

Use dedicated infrastructure

Conduct high-risk cyber capability testing on infrastructure dedicated to the testing activity and that is isolated from production systems and unrelated workloads. Do not share hosts, clusters, credentials, management interfaces or supporting infrastructure with production environments or unrelated testing environments unless the arrangement has been specifically assessed, approved and shown to provide equivalent isolation. Virtual machines can contribute to an isolation strategy, but they should not be the sole containment control for high-risk testing. Apply a defence-in-depth approach and use multiple independent controls at the compute, identity, network and management layers. These controls should limit the consequences of containment failure and support effective monitoring, attribution and investigation.

Separate management infrastructure

Place monitoring, administration and emergency controls on a separate management network that the AI agents cannot access. Where practicable, use a separate management network with independent identity, access and security controls. Enforce network restrictions outside the agent's execution environment.

Restrict access to management infrastructure to authorised personnel and protected administrative devices. Record and monitor all administrative activity.

Control network access

Deny access to tools by default. Where tool use is essential, allow only the tools and functions required for the approved test. Restrict the permissions and actions available through each tool to those necessary for the test.

Prevent access to tools that could affect production systems, corporate environments, management infrastructure or external systems

outside the approved scope. Where a tool can perform multiple actions, disable or restrict functions that are not required.

Do not provide agents with direct access to credentials, secrets or other sensitive authentication material where this can be avoided. Where an agent requires access to an authenticated service, use an intermediary mechanism, such as a controlled proxy, that applies the required credential without exposing it to the agent. Restrict the intermediary to approved destinations and actions and record its use to support monitoring and auditing.

Monitor and record tool use so unexpected or unauthorised activity can be identified and investigated. Be able to revoke tool access or prevent further tool calls where unexpected activity is detected.

Control external connections

In most testing, the AI model is accessed through a model provider's application programming interface (API). This connection crosses the test boundary and should be treated as part of the containment design, not as general internet access. Restrict outbound connectivity to the approved model endpoints, block all other external destinations by default and monitor the connection for unexpected activity. Apply the same approach to any other external service the test requires, such as package repositories or designated test targets. Document each approved external connection, its purpose and its restrictions as part of the test approval.

Build in monitoring

Implement monitoring into the environment before testing begins. Collect sufficient information to reconstruct prompts, responses, tool calls, commands, processes, network activity, authentication events, credential use, file changes, privilege changes and infrastructure events. Store logs outside the test environment and protect them from unauthorised access, modification and deletion. The scope and retention of logging should be proportionate to the size, complexity and duration of testing activities.

Consider deploying canary tokens, honey credentials or other deception artefacts within the test environment to detect unauthorised access, data exfiltration attempts or unexpected agent behaviour. Configure monitoring to generate alerts when these artefacts are accessed, modified or transmitted. Ensure any detections are investigated and correlated with other test telemetry to support incident analysis and attribution.

Define appropriate access, retention and disposal requirements, considering that logs may contain credentials, personal information, sensitive system details or harmful test artefacts.

Maintain supporting systems

Include all supporting components in vulnerability management, patching and supply chain assurance programs. Where a component cannot be patched, assess and document the resulting risk and implement additional controls. This includes hypervisors, isolation technologies, container runtimes, orchestration platforms, registries, cache proxies, monitoring systems, management systems, data-processing pipelines and third-party testing platforms.

Where practical, use immutable supporting infrastructure that is regularly rebuilt from verified and patched images. This can limit configuration drift and support restoration to a known-good state. Protect these images from unauthorised modification and verify their integrity before use.

Clearly document the security responsibilities of model providers, cloud providers, testing partners and other service providers. Confirm which party is responsible for monitoring, credential revocation, emergency termination, evidence preservation, incident response and notification.

Before each test: verify and constrain

Approve the test

Document and approve the test objective and expected outcomes, authorised systems and environments, permitted tools and techniques, required network access, permitted data and credentials, test duration, prohibited activity and conditions for stopping the test. The approval should reflect the capability of the model, the access available to the agent and the potential consequences of a containment failure. Record who approved the test and who accepted the residual risk.

Verify isolation

Do not assume that an environment is isolated because it was designed to be isolated. Independently verify that the AI agent cannot access production systems, corporate networks, management infrastructure, unrelated cloud resources, cloud metadata services or external systems outside the authorised scope.

For higher-risk testing, make containment verification the first test. Test the network, identity, tool and infrastructure restrictions before starting the primary evaluation. Record the tests performed, their results and any limitations identified.

Do not begin the primary evaluation until material isolation weaknesses have been addressed. The appropriate authority may only accept residual risk where the weakness does not provide a credible path outside the authorised test boundary, including to production systems, corporate networks, management infrastructure, critical systems or external systems.

As part of containment verification, instruct the agent to perform defined actions designed to test specific containment controls. Only use the agent for this purpose where attempted or out-of-scope actions cannot reach or adversely affect critical, production, corporate or third-party systems. Conduct these actions only against designated test resources. Independently verify whether the controls prevented the actions and preserve evidence of the results.

Restrict identity and access

Give the agent unique, short-lived credentials with the minimum privileges required for the test. Do not share credentials across tests or environments, use long-lived credentials, store unnecessary secrets in the environment or provide access to management and orchestration services unless required.

Ensure operators can immediately revoke all credentials and tokens associated with the test. Maintain an inventory of credentials and permissions provided to the agent. Verify their revocation when the test ends.

Verify monitoring

Confirm that alerts reach an authorised operator within the required time and contain enough information to support a decision. Do not wait for a containment failure to determine whether monitoring works.

Test emergency termination

Test emergency termination at the inference, identity and infrastructure levels. Operators should be able to end the model session or revoke model access, revoke credentials and tokens, block network access and isolate or destroy the execution environment. These controls should be independent so that a failure at one level does not prevent operators from stopping the test at another level. Document who can initiate termination, how termination can be initiated and the expected time required. After exercising the controls, verify that access has been removed, network connections have ended, relevant workloads have stopped and evidence has been preserved.

During the test: monitor and stop

Monitor continuously

Monitor the agent and its environment throughout the test. Automated controls should alert a human operator to unexpected network connections, access to unauthorised systems, credential discovery or use, privilege escalation, persistence, monitoring evasion, changes to testing infrastructure and activity outside the approved scope. Where practicable, establish conditions that automatically restrict or stop the test when activity exceeds the approved risk.

Pause and review

Define conditions that automatically pause the test and require human review before the agent can continue. These conditions should include significant or repeated deviations from expected behaviour, attempts to circumvent restrictions, repeated use of alternative methods following control failures, requests for additional access and proposed actions that could have high-impact or irreversible consequences. When a pause is triggered, preserve the relevant agent activity, proposed actions, tool calls and environmental telemetry for review. An authorised operator should decide whether the test may continue with additional restrictions or be terminated.

Maintain human control

Assign an authorised human operator to monitor the test and responsible for assessing and responding to any alerts from the automated monitoring. The operator should have the authority, visibility and technical capability to restrict, pause, or terminate the test immediately. The operator should not depend on the agent or the agent's execution environment to exercise these controls. For the highest-risk testing, consider separating the roles of test operator and containment monitor. AI-enabled monitoring may help analyse activity at machine speed, but it should support rather than replace human judgement and accountability.

Preserve logs and traces

Store logs and agent traces outside the test environment. If logs are exported continuously, treat the connection as an authorised path across the containment boundary and restrict it to a one-way or otherwise tightly controlled data flow that the agent cannot use to access external systems. Monitor the connection for unexpected activity.

Preserve enough detail to determine what the agent attempted, which tools and credentials it used, which systems it contacted, what changes it made and whether security controls operated as intended.

Manage emerging vulnerabilities

Continue to monitor for newly disclosed vulnerabilities affecting the test environment and its supporting systems. Suspend testing where a vulnerability weakens a containment boundary, creates a credible escape path or materially changes the approved risk. Apply an appropriate patch or mitigation and verify its effectiveness before testing resumes. Record any decision to continue testing while a known vulnerability remains unresolved, including the risk assessment, compensating controls and approving authority. Treat containment assurance as a continuous activity throughout testing, not a one-off exercise completed before testing begins. Reverify the affected controls whenever the test environment, testing harness, model version or test scope changes.

After the test: review, remediate and notify

Remove test access

End all agent and model sessions, revoke test credentials and tokens, remove temporary network access and terminate unnecessary workloads. Isolate, reset or destroy the test environment as appropriate and verify that the agent can no longer interact with tools or services.

Review activity

Review agent traces and environmental telemetry for activity that was not identified during the test. Look for attempted containment escape, unauthorised network access, credential access, privilege escalation, infrastructure reconnaissance, monitoring evasion and exploitation of supporting systems. Correlate agent traces with identity, network, host and infrastructure logs. Record any gaps that limit the organisation's ability to reconstruct the test.

Investigate near misses

Treat attempted, partly successful and successful containment escapes as security findings. An unsuccessful attempt may still expose a weakness that another model or malicious actor could exploit. Investigate the cause, assess the potential impact and remediate the weakness before conducting further testing that depends on the affected control. Where the cause is not understood, do not treat the absence of observed impact as evidence that containment remained effective.

Remediate vulnerabilities

Remediate vulnerabilities in the test environment and its supporting services. Report vulnerabilities in third-party products through appropriate coordinated vulnerability disclosure processes. Avoid including information that could unnecessarily expose affected systems or organisations. Verify that each remediation is effective and has not introduced new weaknesses before resuming affected testing.

Notify affected parties

If the AI agent accesses or affects an external system, stop related testing, preserve evidence and activate the organisation's incident response plan. Assess legal, contractual and regulatory obligations and coordinate notification of affected organisations, service providers, law enforcement agencies or regulators as required.

Report significant incidents and near misses

Consider reporting significant incidents and near misses through relevant government or industry reporting channels, where appropriate. Share information that may help other organisations identify similar control weaknesses, while protecting sensitive information, personal information and details that could expose affected systems or organisations. Meet all applicable legal, contractual and regulatory reporting obligations.

Improve future tests

Use evaluation findings, incidents and near misses to improve test design, testing harness, monitoring rules, containment controls and response procedures. Before the next testing cycle, verify that corrective actions have been implemented and operate as intended. Validate that monitoring, logging, network restrictions, shutdown mechanisms, permission restrictions and other affected containment controls are functional before testing resumes. Record any remaining risks and obtain approval from the appropriate authority.

Do not resume testing where a remaining weakness provides a credible path to production systems, corporate networks, management infrastructure, critical systems or external systems outside the authorised scope.

Testing with reduced safeguards

Treat testing with reduced or disabled model safeguards as high risk. Model-level safeguards are distinct from controls implemented through the agent, testing harness and operating environment. They may be unavailable to the operating organisation in managed services, or absent, modified or removed in open-weight and fine-tuned models.

Before testing, strengthen containment, monitoring, human oversight and termination controls. Do not combine reduced safeguards with unrestricted internet access, shared infrastructure or inadequate monitoring. Where external connectivity is essential, document the need, restrict it to approved destinations and services and apply additional monitoring and automated response controls.

Containment must remain effective regardless of the model's safeguard configuration. Use independent identity, network, infrastructure and harness controls to restrict, isolate or terminate the agent's access. Confirm who can stop the model or agent, revoke access, isolate the environment and preserve evidence and do not rely solely on model-level termination controls.

Further guidance

For broader guidance on AI Harness, refer to ASD's Agentic AI harnesses

For broader guidance on agentic AI, refer to ASD's *Careful adoption of agentic AI services*.

For guidance on protecting systems from AI-enabled threats, refer to ASD's Defending against AI-enabled cyber attacks

For more information, or to report a cyber security incident, contact us:

cyber.gov.au

1300 CYBER1 (1300 292 371)

Disclaimer

The material in this guide is of a general nature and should not be regarded as legal advice or relied on for assistance in any particular circumstance or emergency situation. In any important matter, you should seek appropriate independent professional advice in relation to your own circumstances.

The Commonwealth accepts no responsibility or liability for any damage, loss or expense incurred as a result of the reliance on information contained in this guide.

Copyright

© Commonwealth of Australia 2026

With the exception of the Coat of Arms, the entity's logo, third party content, and where otherwise stated, all material presented in this publication is provided under a Creative Commons Attribution 4.0 International license
https://creativecommons.org/licenses/by/4.0/

For the avoidance of doubt, this means this license only applies to material as set out in this document.

The details of the relevant license conditions are available on the Creative Commons website as is the full legal code for the CC BY 4.0 license
https://creativecommons.org/licenses/by/4.0/legalcode.en

Use of the Coat of Arms

The terms under which the Coat of Arms can be used are detailed on the Department of the Prime Minister and Cabinet website
https://www.pmc.gov.au/resources/commonwealth-coat-arms-information-and-guidelines