Presenter Script: AI Agent Kill Switch Workshop

Speaker notes for the kill switch fact file, section by section

Companion to: AI Agent Kill Switch Fact File v1.0 **Role:** Security architect, leading the session **Date:** November 2026

Use this alongside the fact file. For each section it gives the point to land, a few things to say in your own words, background to reach for if someone digs in, and the question most likely to come back at you. The three things to keep returning to: a kill switch is three actions not one, everything needed is generally available today, and the two caveats (the token gap and the egress bypass) are real and you are naming them, not hiding them.

> **HOW TO OPEN THE SESSION**
>
> **Say:** "The question today is not whether we adopt agents, we are licensed for Agent 365. It is how we make sure we can see one, govern one, and stop one before it does damage. I will walk through what a kill switch actually is on this platform, what we can build today versus what is still in preview, and the handful of decisions we need to leave here with." Set the tone as practical and honest: you are here to enable agents safely, not to block them.

Section 1, what a kill switch actually is

The point: there is no single button. Stopping an agent is three coordinated actions.

SAY

- Stopping an agent means three things together: stop the runtime in Foundry, disable the identity in Entra, and cut the network path. Any one on its own leaves a gap.
- Runtime-only leaves the identity live, so it can be re-invoked. Identity-only leaves a running process holding a valid token until it expires. Both, plus a network cut, is what actually stops it.
- In an incident the order is: disable the identity, stop the runtime, confirm the network is cut.

BACKGROUND

This reframes the room's likely assumption that there is a stop button or that deleting the agent is enough. It is the foundation for everything after, so land it clearly before moving on.

> **Likely question:** "Isn't deleting the agent enough?"
>
> No. Deleting the Foundry agent stops it serving, but it does not disable the Entra identity, so the same identity could be used again. That is why the two are separate steps.

Section 2, the three mechanisms

The point: here is exactly how each of the three actions is done, and the one caveat that catches everyone.

SAY, IDENTITY (2.1)

- Disabling the identity has three scopes, surgical to blunt: one agent, a whole blueprint (every agent of that type at once), or tenant-wide with a Conditional Access block.
- Disable is reversible, so it is the containment action. Delete is permanent, so that is retirement, not a kill switch.

SAY, THE TOKEN GAP (THE DIAGRAM)

- This is the caveat to be honest about. Disabling blocks new tokens immediately, but a token already issued keeps working for up to about an hour, unless the resource supports Continuous Access Evaluation, which revokes in minutes.
- So the mitigation is: put sensitive resources on CAE, and revoke sessions. And it is a reason the network cut matters as a backstop.

SAY, RUNTIME (2.2) AND NETWORK (2.3)

- The runtime kill is a delete with force, which tears down active sessions, not just the registration.
- The network cut removes the agent subnet's egress and private access while the first two propagate.

BACKGROUND

Identity disable is a Graph call setting `accountEnabled:false`; the agent user, if it has one, must be disabled separately. CAE becomes mandatory for new tenants from 1 October 2026. Hosted agents are containers, published as immutable versions.

> **Likely question:** "So we cannot actually stop it instantly?"
>
> On CAE-capable resources, effectively yes, within minutes. On resources that are not CAE-capable, there is a gap of up to the token lifetime. That is exactly why we will require CAE on sensitive resources and keep the network cut as a backstop. It is a known residual risk, not a surprise.

Section 3, licence to operate

The point: prove the kill switch earlier in the pipeline before it ever runs in production.

SAY

- We progress through dev, pre-production, production, and the kill switch has to be proven at each gate before an agent promotes.
- Dev: the manual kill works end to end and telemetry flows. Pre-production: the automated kill works with an approval gate, and we have measured how long it takes. Production: we exercise it on a schedule and keep the evidence.
- The rule: an agent does not move up until its kill switch has been exercised and evidenced at the current stage.

BACKGROUND

This is the governance hook. It turns the kill switch from a design claim into something tested, and it gives the board a clear gate to point at.

> **Likely question:** "Isn't a documented playbook enough to sign off?"
> No. A document is a plan. The evidence we want is "we ran it last Tuesday and it stopped the agent in N minutes." That is the difference between assuming the control works and knowing it does.

Section 4, grading the kill switch

The point: not every agent needs the same kill switch. Match it to privilege.

SAY

- We grade agents low, medium, high by what they can reach and do, and the kill switch scales with the grade: how fast it acts, how much is automated, and whether containment happens before or after human approval.
- A high-privilege agent, one that can write to systems or move money, justifies auto-containment with approval only to restore. A low-privilege agent that reads public data does not.

BACKGROUND

This is the privilege-spectrum idea: spend the control effort where the risk is. It also stops the opposite failure, wrapping heavyweight controls around trivial agents and slowing adoption.

> **Likely question:** "Who decides an agent's grade?"
> That is one of our decisions to close today, part of the ownership model. The owner proposes the grade at creation; security validates it. It should be set before the agent goes anywhere near production.

Section 5, what triggers a kill switch

The point: triggers come from identity signals, behaviour, and people, and the good ones express intent drift.

SAY

- Three sources: identity risk signals from Entra, behaviour and intent signals from telemetry in Sentinel, and manual triggers from an owner or the SOC.
- The important distinction: a good trigger says the agent is heading somewhere it should not, not just that it did something. That is what keeps the switch trusted rather than noisy.

BACKGROUND

Entra ID Protection detects offline risk types for agent identities (unfamiliar resource access, sign-in spikes, failed access). The behaviour signals come from the telemetry in sections 8 and 9.

> **Likely question:** "Won't this fire constantly and stop legitimate work?"
> That is why we grade it. Low-tier agents alert and a human decides. Only high-tier agents auto-contain, and only on high-confidence triggers, with approval to restore. We tune for intent drift, not raw activity.

Section 6, who has the authority to kill

The point: define who can pull the switch at agent level and at platform level, before go-live.

SAY

- Two levels. Agent level, one agent or type, sits with the named owner and the SOC on-call, who hold the Agent ID Administrator role and the Foundry project role. Platform level, tenant-wide, stops every agent at once and needs a senior, named authority and a logged decision.
- The decision for this room: name the actual people for each level, and agree the approval chain for auto-contain and for restore.

BACKGROUND

Least privilege applies to the kill switch itself: the people who can disable agents should be a named, small set, not a broad group.

> **Likely question:** "Who can pull the platform-level switch, and when?"
> That is a decision we need to make today. It stops all agents, so it should be a named senior authority acting under a declared condition, for example a confirmed compromise, with the decision logged.

Section 7, network, ingress and egress

The point: only private networking gives us the egress cut, and the egress firewall is not a complete control.

SAY

- Two postures. Basic has public egress and is not suitable for classified data. Standard Setup with private networking keeps agent traffic off the public internet and gives us private endpoints and an egress firewall. We should mandate Standard for anything above low tier.
- Be honest about the caveat: some tool calls are made through an internal mechanism that bypasses the egress subnet, so the firewall does not see every outbound action. That means identity and runtime controls carry more of the weight, not less.

BACKGROUND

Standard setup needs bring-your-own VNet and bring-your-own Storage, AI Search and Cosmos DB with private endpoints, which are not auto-created. The container registry for hosted-agent images generally has to stay reachable. Private networking now extends to tool connectivity (MCP, AI Search, Fabric).

> **Likely question:** "So egress is fully locked down?"
> Not completely, and I want to be straight about that. The egress firewall covers the subnet, but some platform tool calls bypass it. We compensate by keeping agent identities tightly scoped and the runtime killable, so even an uncontrolled egress path has little to carry.

Sections 8 and 9, observability and the OpenTelemetry route

The point, and the headline of the whole session: the telemetry we need is generally available today, we do not wait for the preview connectors.

SAY

- We cannot trigger or justify a kill switch on behaviour we cannot see. Foundry emits agent telemetry through OpenTelemetry into Application Insights, automatically for hosted agents, capturing model calls, tool calls and the agent id.
- The key point: the native Agent 365 connectors into Sentinel are still preview, but we do not need them. OpenTelemetry to Application Insights to a Sentinel-enabled workspace is generally available. We build on that now, and treat the connectors as a later enhancement.
- Walk the flow diagram: agent to App Insights to Log Analytics to Sentinel, where the analytics and the kill-switch playbook live.

BACKGROUND

App Insights is backed by Log Analytics, so pointing it at the Sentinel workspace puts agent traces in standard tables where Sentinel can act on them. Correlate on the agent id, model, tool call, session and the identity the agent acted for.

> **Likely question:** "Shouldn't we wait for the connectors to go GA?"
> No. Waiting would delay the whole observability and kill-switch capability for a richer version of something we can already do. We build on the GA OpenTelemetry path now and add the connectors when they are GA, with no rework of the core.

Section 10, SOC playbooks and runbooks

The point: this is how the kill switch runs in practice, automated and manual.

SAY

- Walk the sequence diagram: a Sentinel analytic fires, a playbook runs, an approval step for medium-tier agents, then the playbook disables the identity, deletes the runtime, and cuts egress, and logs it. Restore needs a separate approval.
- The manual runbook is the fallback and the platform-level path: preserve evidence first, then disable, delete, confirm, and decide whether to re-enable, re-scope or retire.

BACKGROUND

The automated path is a Sentinel Logic App playbook calling Microsoft Graph and Foundry. Evidence capture first matters for the investigation and for any audit of the containment decision.

> **Likely question:** "What if the automation fails?"
> That is what the manual runbook is for, and why the SOC on-call holds the roles to do it by hand. The automation is the fast path, not the only path.

Section 11, what is possible today versus roadmap

The point, say it plainly: every component of a working kill switch is GA today.

SAY

- Identity disable, runtime delete, private networking, OpenTelemetry to Sentinel, and the SOAR playbooks are all generally available. We can build and operate the kill switch now.
- The preview items, the native connectors, Defender agent detection, client-side tracing, make it richer and easier, but they are not on the critical path.

BACKGROUND

This is the slide that answers the "is this real yet" question. Keep it factual, point at the table, and note that preview status moves, so we reconfirm before build.

> **Likely question:** "How much of this is vapourware?"
> Very little. The table splits it explicitly. Everything in the GA column is live and documented today; the preview column is additive. If anything, the risk is the preview features changing, not the GA ones being absent.

Section 12, decisions to close

The point: do not leave the room without owners against these.

SAY

- Read the seven decisions out, and for each one ask for an owner and a date, not a discussion. Private networking mandate, grading model, named kill authority, CAE on sensitive resources, the OpenTelemetry route, the egress-bypass compensation, and the game-day cadence.
- Close by restating the three things: a kill switch is three actions, it is buildable today, and the two caveats are named and mitigated.

> **Likely question:** "What do you need from us to start?"
> Agreement on the private networking mandate and the grading model, and named people for the kill authority. With those, we can stand up the kill switch in a development stack and prove it before anything touches production.

THREE THINGS TO BE HONEST ABOUT, DO NOT OVERSELL

1. The token gap: disable is not instant on non-CAE resources. **2. The egress bypass:** the firewall does not catch every outbound tool call. **3. Preview status:** the connectors and Defender agent detection are preview and can change. Naming these builds trust in the room; glossing over them is what gets found out later.

Presenter script, companion to the AI Agent Kill Switch Fact File v1.0, November 2026. Talking points are a guide, say them in your own words. Reconfirm preview status before any build decision.