THE PROBLEM

A maintenance window, and no reliable map.

An out-of-date list, a tool that cannot tell devices apart, and an attack route that exists only on a whiteboard.

Monday. The company has a maintenance window on Saturday. The work order names a critical system, and an outside supplier wants remote access to a maintenance computer so they can update it.

Before anyone can approve that, the questions pile up. Which devices are actually on that network today? Which of them have known weaknesses? If someone broke into the maintenance computer, what could they reach from there?

The asset list is a spreadsheet from last year. The tool that checks for weaknesses cannot tell one camera from another. And nobody has drawn the path from the office network to the critical systems; it lives on a whiteboard, if anywhere. So the meeting turns into an argument about whose list is right, instead of a decision about Saturday.

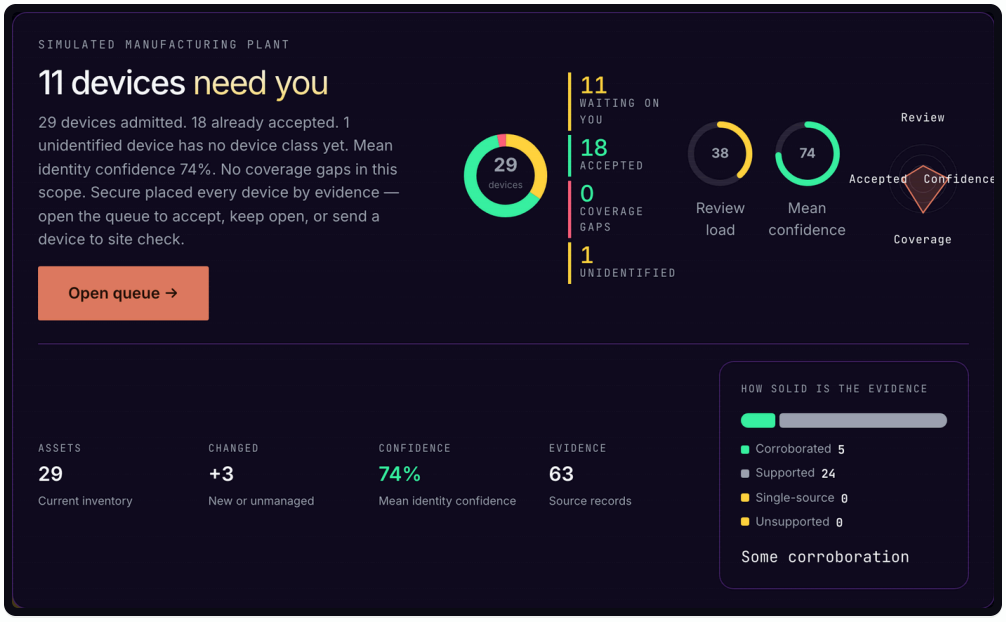
What the team needs is a list of devices built from evidence, with the uncertain ones clearly marked, known weaknesses that stay “possible” until someone confirms them, and a route they can walk through step by step. That is what Secure builds, and the next three pages follow the week through it.

WHAT SECURE FOUND

The devices it could hear, each with its evidence.

How what the collector hears becomes a device list the team can defend.

These screens come from the practice company network built into Secure.



The first screen: 29 devices, 18 confirmed, 11 waiting for a decision (one of them not yet named).

Secure starts with a collector: a small piece of listening software placed on the company network. It reports what it hears, and Secure turns that into a list of devices, giving each an identity and a confidence score based on the evidence behind it. In the practice company network it found 29 devices, and on average it was 74% sure of each identity.

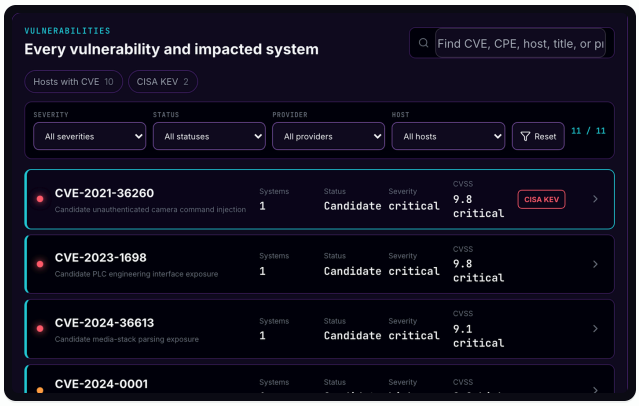
It does not hide its doubts. Eighteen devices have been accepted by a person. Eleven wait in a review list, where an engineer can confirm each one, leave it open, or send someone to look at it in person; one of those could not be named at all, and the screen says so. Even so, the list only covers what the collector could hear.

Some checks mean contacting a device directly rather than just listening, and that can disturb equipment that controls physical processes. Secure only does it with signed permission, and it treats industrial control equipment as safety-critical.

WHAT IT MEANS

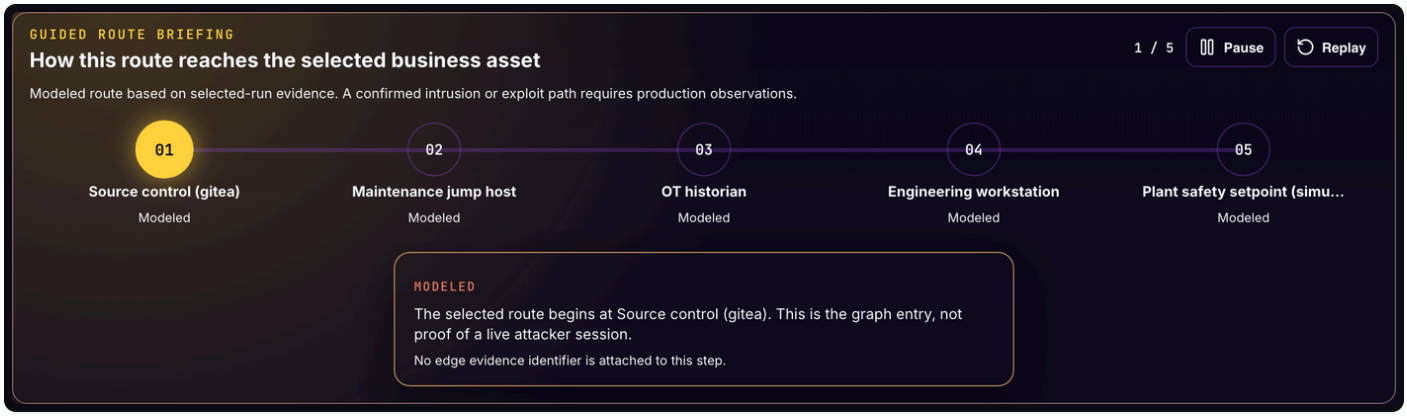
Flaws that stay “possible”, and a route you can inspect.

Known flaws, what they do and do not prove, and how an attacker could reach critical systems.



Secure compares each device’s software with a public database of known flaws, kept on your own computer so no internet is needed. In the practice company network, ten devices had possible matches, and two are on the US government’s list of flaws that attackers are already using.

The most serious is a possible flaw in a security camera. If a person confirms it, someone could send that camera commands without logging in. Until then it stays **possible**, and the screen says why: a match needs a person to confirm it, and it does not prove anyone can break in.



A modeled route from the code server to a safety control. Every step is marked “Modeled”.

Secure also maps how an attacker could move from system to system. The route above starts at the software team’s code server, passes a maintenance computer, an operations records server and an engineer’s workstation, and ends at a safety control. Every step is marked as modeled: it is a plan to review, not evidence of an attack.

This is where Saturday’s decision gets easier. Before anyone gives the supplier remote access, Secure can run that change on a simulated copy of the network and show what else it would touch. The copy is a model, not your live network, and your team still makes the change.

RUNNING IT AT YOUR SITE

The same week, with Secure in place.

What changes for the team, what Secure needs, and where it stops.

Picture Monday's meeting again, with Secure in place. Instead of arguing over spreadsheets, the team opens one list of devices, each with the evidence behind it and the uncertain ones marked. The eleven devices waiting for a decision get confirmed or sent for a check. The possible flaws on those devices are confirmed or dismissed by the people who look after them. The route from the office network to the critical systems is on screen, step by step. Saturday's approval is made on that.

To get there, Secure needs a collector on each network you approve, placed where it can hear the traffic that matters. Someone on your team starts each survey, and you can compare two surveys to see what changed; it does not run on a schedule. Some checks contact a device instead of only listening, which can disturb a running line, so Secure requires signed permission for them; agree with the people who run the line how their equipment is watched today before anything is connected.

Secure cannot promise it has found every device: it reports what its collector could hear from where it was placed. Information about cloud services is imported from your provider, not discovered. Secure has no built-in AI, although your own AI assistant can connect to it if you choose. It can run on your own computers, with its list of known flaws held locally; QASO also runs a hosted version, which uses Google sign-in for accounts an administrator has approved. Ask which one you would run.

Planning for quantum computers fits the same way. When a service opens a secure connection, the two sides first agree on secret keys; Secure records which method they agree to use, for connections that use today's standard. It does not look at security certificates or older kinds of connection, so it is not a complete list of your encryption. You tell it how long your data must stay secret, how long a changeover would take and when you expect quantum computers to become a threat, and it shows which of your options leaves the fewest of those dates at risk. It does not make the change.