Proof & Evaluation

Evidence, testing, and evaluation guidance - documented results, how to run a meaningful proof of concept, and what to ask any prevention vendor.

- Does Cyber Crucible pass ransomware simulation tests?
- Does Cyber Crucible have false positives? What is the false positive rate?
- Does Cyber Crucible test all possible ransomware exhaustively?
- What evidence is there that Cyber Crucible actually stops attacks?
- How should we run a proof of concept for a prevention tool?
- What questions should we ask any endpoint security vendor?
- What happened to the security industry's "dumb collector" model?
- Why does intent-based prevention generate less alert noise than signature-based detection?

Does Cyber Crucible pass ransomware simulation tests?

The best answer is…it depends on the quality and accuracy of the ransomware simulations, but we haven’t seen many high quality tests that match true attack tools and behaviors.

Cyber Crucible examines underlying behavioral patterns in file access, memory behaviors, process behaviors, and cryptographic behaviors to identify and suspend data extortion activities in conjunction with an attack.

Some simulations of data extortion software have evolved over time, to be a more accurate representation of real attacks.

We have experienced simulations that, appeared to focus on checking for the data extortion countermeasures disclosed by some vendors, and not the tradecraft of the extortion tools and attackers themselves. A way to say this may be, “Testing for the countermeasure, not for the attack”.

In response, we never shy from tests against attacker emulation, penetration tests, or extortion tools. We routinely attack our own software, and replay attacker tradecraft we observe discussed online or found (and stopped) in client environments.

Does Cyber Crucible have false positives? What is the false positive rate?

Cyber Crucible strives for a 0 false positive product environment. Having said that, there are sometimes false positives. Let’s discuss where these come from.

Currently, the false positive rate is approximately 1 response per month, per 1800 deployments/agents, with full accountability for why those responses occur.

Corrupted Program Doing Rapid File Operations

Whether on purpose or through poor programming, sometimes programs either corrupt themselves, or are corrupted by other programs while running. We see this a lot with new features on large program suites, such as Microsoft Office, or proprietary custom-built software.

What happens is that the program’s memory is corrupted, which increases the level of inspection in the program. If that is then followed by extortion-like file access behavior, Cyber Crucible is forced to suspend the program. We’re getting better at identifying known bugs, and you get to at least run your program until it crashes (some automatically restart, some do not).

In a world where attackers do not use the file system, and hijack the memory of running programs instead, we have to err on the side of caution.

Cyber Crucible now has the ability to create a permission, where, if Program A, with arguments Arguments A, causes a corruption in Program B, with Arguments B - we can create a temporary exclusion for:

(Program A + Arguments A) opens (Program B + Arguments B)

That way there isn’t additional risk assumed that a hacker is involved, unless they just happen to use the exact same arguments.

An excellent example of this was when Microsoft first introduced opening Office documents from within their desktop chat software. It corrupted the Office program…every…single…time….then the Office program would start scanning all Documents on the machine about 25% of the time. That…needed an exclusion.

If you experience a crash, use the support button, and we'll help you make an exclusion, until the vendor fixes the issue.

Security Programs

Cyber Crucible does really well co-existing with security programs. Opposite to what most new customers think, more advanced security tools tend to use more advanced system interactions than antivirus programs that use older technologies. In fact, some free or inexpensive security tools run a variety of insecure Powershell scripts as a key part of their protection, versus actual programs.

As a highly resilient kernel-based security tool, Cyber Crucible is best viewed as being "lower", or "closer" to the hardware than most other tools. Cyber Crucible automatically identifies and adds other security tools to the behavioral models. As long as the security tools don't show signs of being compromised, they are allowed to operate normally.

Occasionally, a security tool will cause system instability, when their attempts to shut off or interfere with Cyber Crucible software fail. In those cases, it is best to whitelist CyberCrucible in the other security tool, to prevent it from trying (and failing) to disable or interfere with Cyber Crucible.

Feel free to open a support ticket with Cyber Crucible, via the web portal, to discuss the matter.

Administrative or Backup Tools

In some circumstances, a tailored behavior must be created for an administrative or backup tool, while the Cyber Crucible team develops an improved behavioral model. In this case, the Cyber Crucible team will assist in the creation of an exception in the behavioral model, which will single out the program plus arguments for an extremely small window of opportunity for a possible attack.

Does Cyber Crucible test all possible ransomware exhaustively?

Cyber Crucible operates off of kernel level behavioral modeling, to discover data theft, credential theft, and ransomware encryption behaviors very quickly. The behavioral decisions are made using information from process behavior, memory-derived behaviors, and certain types of file-sourced behaviors, to make a decision right as the extortion is about to occur.

Despite the very large number of ransomware and extortion-themed software samples, they can be behaviorally categorized into a relatively small number of sets.

On the surface, though, “skin-deep” defenses used by a variety of security tools have a very large number of extortion tools to try to counter. It is important to understand our behavioral analytics run much deeper into the attacker tools and tradecraft, dramatically reducing the need for some type of (impossible) exhaustive testing of all possible malware at all times.

The Cyber Crucible developers categorize the kernel level memory, process, and file behaviors of an extortion tool, then ensure it fits into one of the known defensive capabilities. Occasionally, similar to a vaccine, the formula can be tailored slightly to produce a more accurate response.

Very rarely does something completely new appear.

The Cyber Crucible team works exhaustively to preemptively uncover novel techniques that are not yet observed in our customer base, or in threat intelligence.

The effect?

1. Cyber Crucible’s extortion tool defense is complete against all known ransomware variants.

2. New ransomware variants are blocked before they even hit the first customers. We don’t even know what to call most of the software we defend for around 120 days, until researchers “catch up”.

3. Ransomware and extortion tool developers sometime call us to see what we’re up to, to try to work out an angle. Unfortunately, frustration from our presence appears to sometimes spur evolution on their part, which makes other tools even less effective.

4. We welcome testing of our software. When penetration testers, malware analysts, or adversary emulation professionals enter the sale process - we get excited!

What evidence is there that Cyber Crucible actually stops attacks?

Short answer: Documented customer deployments, independent testing by a major accounting and advisory firm, and repeated instances of stopping zero-day attacks months before other vendors reported them.

Documented outcomes

- **Financial services:** nearly 10,000 malicious processes autonomously intercepted in 60 days, 98% on a highly privileged Microsoft SQL server farm — with no alerts from three deployed EDRs, an MDR, or an outsourced Top-50 SOC.
- **Independent testing:** one of North America's largest accounting and advisory firms put Cyber Crucible through its own evaluation against ransomware, data theft, and identity fraud scenarios.
- **Ahead of the industry:** Cyber Crucible has stopped multiple zero-day attacks on customer networks up to 90 days before any other vendor reported or responded to them.
- **Browser-based attacks:** from Q4 2023 onward, automated responses against Chrome, Edge, and Chromium activity rose from zero into the thousands, with related CVEs later published by Google and Microsoft.
- **Ransom payments:** roughly 90% of ransomware victims paid in 2023. Cyber Crucible customers paid $0.

Resilience under direct attack

In maritime deployment, adversaries escalated to hijacking Windows credentials and locking systems at the BIOS level when conventional attempts failed. With two other customers, attackers who couldn't defeat the kernel decision engine tried instead to block customer notification by attacking the OS networking stack — and failed, because Cyber Crucible's network communications are independent of the operating system.

How should we run a proof of concept for a prevention tool?

Short answer: Deploy it alongside your existing stack in a real production environment and compare what each tool sees. The most informative result isn't a lab test — it's the gap between what your current tools alert on and what actually gets intercepted.

Why deploying alongside works best

The financial services firm in our best-documented case did exactly this. They didn't remove anything. They added Cyber Crucible specifically to find out what their existing investment was missing, and had a definitive answer within 60 days.

What to measure

- **Interceptions your current stack never alerted on.** This is the core signal.
- **Time to decision.** Prevention has to complete before damage, not before a ticket is closed.
- **Business disruption.** Count downtime, reboots, and false-positive-driven interruptions.
- **Analyst hours consumed.** Autonomous prevention should reduce workload, not add a queue.

A caution about lab testing

Testing against samples pulled from malware databases often measures very little, because attackers design malware to stay dormant unless it receives a validation signal from a live command-and-control server they still operate. A sample that does nothing in your lab proves nothing about a live attack.

What questions should we ask any endpoint security vendor?

Short answer: Ask where decisions are made, what happens without connectivity, what the tool depends on to function, and whether it prevents or merely reports. The answers separate architecture from marketing quickly.

Questions worth asking

1. **Where is the protective decision made — on the endpoint or in your cloud?** A network round trip in the critical path cannot beat a millisecond-scale attack.
2. **What happens when the machine is offline or air-gapped?** If protection degrades, it wasn't local.
3. **Does your agent depend on Windows system libraries to function?** If yes, an attacker who compromises those libraries in memory can blind or silence it.
4. **Do you require signatures or threat intelligence feeds?** If yes, protection trails the attacker by definition.
5. **Does a response require a human?** If yes, response time is bounded by human biology.
6. **Do you store our encryption keys or upload sensitive data for analysis?** Centralized storage creates a target and a compelled-disclosure risk.
7. **What is your false positive rate, and what does a response actually do?** Full-system lockdown as the only containment option is expensive in a hospital or a plant.

Why these questions matter

Most endpoint tools look similar in a datasheet. These questions surface the architectural decisions that determine whether a tool can act in time — or can only explain, afterward, what happened.

What happened to the security industry's "dumb collector" model?

Short answer: For years vendors advised that local endpoint processing was obsolete and pushed lightweight agents that forward raw telemetry to cloud analytics. Modern in-memory attacks against core OS libraries have exposed that model — the agents keep running but go effectively blind and mute.

Why the model was adopted

It was cheaper to build. Kernel-level engineering is difficult and expensive; shipping telemetry to the cloud let vendors ship inexpensive endpoint software, monetize large cloud storage pools, and market cloud AI capabilities. The model optimized for development economics, not defense.

Why it's failing now

These agents depend entirely on native operating system components to function and gather data. When attackers compromise those components in memory, the agent doesn't crash — it stays running and reports nothing, leaving a dashboard that says everything is fine while data is exfiltrated.

There's also documented deliberate interdiction: attackers, and in some verified instances competitors seeking to mask their own flaws, actively exploiting these library dependencies to force-terminate or sabotage endpoint defenses.

The architectural dead end

Fixing this would require legacy vendors to abandon cloud-first telemetry pipelines and rebuild for local, kernel-level edge computing — a multi-year effort. Recent acquisitions and venture investment show the industry moving further toward cloud analytics and centralized data lakes instead.

Why does intent-based prevention generate less alert noise than signature-based detection?

Short answer: Because it asks a narrower question. Signature and heuristic tools alert whenever something *resembles* a known-bad pattern and leave humans to sort it out. Intent-based prevention asks whether a specific program is, right now, doing something malicious at a known data-theft entry point — a question with far fewer ambiguous answers.

> For Cyber Crucible's measured false-positive rate, see the existing article **"Do you have false positives? What is your false positive rate?"** This page explains the architectural reason the noise profile differs.

Why the noise profile is different

Signature and heuristic tools generate alerts when something *resembles* a known bad pattern, then rely on human triage to sort real from irrelevant. That's what produces 100+ alerts per endpoint per day in many environments.

Intent-based assessment asks a narrower question: is this program, right now, doing something malicious at a known identity-theft or data-theft entry point? That question has far fewer ambiguous answers.

What this means operationally

- Analysts aren't chasing low-confidence alerts.
- There's no YARA rule tuning or manual threat hunting required to keep noise manageable.
- Because the response suspends only the offending process, even an incorrect interception does not take down a system.

For current measured rates in an environment like yours, ask your Cyber Crucible representative — and see the existing "Do you have false positives?" article for product-level detail.