Chris Jordan

MIND THE GAP

WHEN PREVENTION FAILS

CONTENT

THE PROMISE AND REALITY OF

It is only natural that organizations invest heavily in prevention. The logic is simple: if a threat can be stopped before it causes harm, there is no need for costly recovery efforts or investigations. Preventing an issue before it materializes appears, at first glance, to be the most efficient and cost-effective strategy for maintaining security. This mindset underpins the core design of many cybersecurity programs today, particularly in mid-sized businesses seeking to strengthen their defenses without overwhelming their teams or budgets.

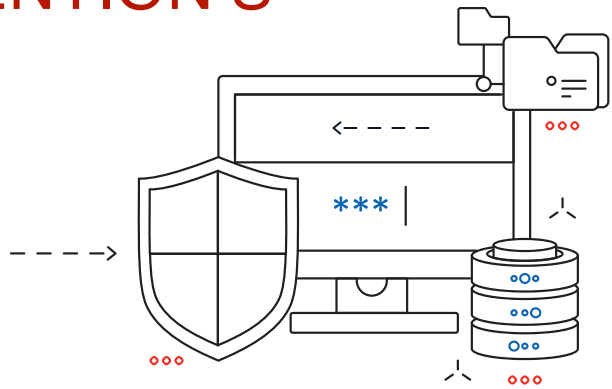
As companies move deeper into building their defenses, prevention often extends beyond the installation of a single product. It becomes a broad effort, encompassing disciplines such as patch management, where systems are regularly updated to remove known vulnerabilities. It also includes the deployment of endpoint detection and response (EDR) tools to monitor and protect desktops, the installation of firewalls to control network traffic, and the implementation of email security gateways to filter malicious communications. Data Loss Prevention (DLP) solutions may also be introduced to protect sensitive information. Collectively, these measures are deployed across the infrastructure to uphold what is often referred to as the organization's security posture.

The term “security posture” captures the overall defensive stance of the company—how its systems are configured, monitored, and managed to enforce protective policies. Ideally, this posture reflects a well-organized, self-regulating system where threats are identified and neutralized automatically, minimizing the need for human intervention. It is a desirable vision: a state in which security is embedded so thoroughly into the environment that continuous oversight seems unnecessary.

However, if prevention alone were sufficient to guarantee security, there would be no need for monitoring teams, incident response plans, or security operations centers. The systems would operate independently, repelling all threats with perfect efficiency. Yet, in practice, no organization—no matter how well-resourced or sophisticated—achieves this. This fundamental reality is where a critical question arises: do you believe that prevention alone is enough? If so, there is little need to read further. But if you recognize that there may be gaps between prevention and true operational security, gaps where threats still find a way to emerge and impact your environment, then understanding the nature of these gaps becomes essential.

This e-book is dedicated to examining that space—the “prevention gap”—where the protections organizations build begin to falter, and where the true demands of security operations come into play. In the pages that follow, we will explore why prevention is never absolute, how to recognize when it fails, and what it takes to bridge the space between prevention and effective response.

UNDERSTANDING PREVENTION'S



This latency—this unavoidable delay in interpreting and determining whether something is truly malicious—is at the heart of a fundamental limitation in prevention technology. When threats are known and their signatures are catalogued, prevention can act almost instantly. But when threats are novel or obscured, the system enters a race condition: it must decide whether to allow or block an action before it has enough certainty. In this race, prevention systems sometimes lose. And even when they eventually identify a threat, the damage may already be underway.

It is tempting to think that race conditions are rare or unlikely, but the opposite is true. Security technologies are constantly racing against attackers' creativity and adaptability. Worse, there are specific reasons why prevention systems, even when aware of a potential issue, will deliberately hesitate to act. First, there is the risk of operational disruption. A classic example is the McAfee 5958 incident, where an antivirus update incorrectly classified a critical Windows system file as malicious.

Cheat sheet to understanding why prevention fails

1

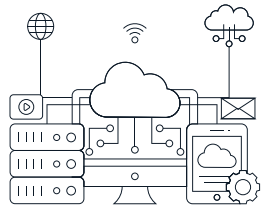
LATENCY FROM UNKNOWN ATTACKS

- Prevention works fast for known threats, but when facing an unknown or modified threat, the system must interpret the behavior before acting.
- This creates a race condition where prevention tools may lose — the attacker acts before the system can make a confident decision.

2

OPERATIONAL RISK (REGRESSION TESTING / FEAR OF BREAKING SYSTEMS)

- Even when prevention detects something suspicious, it may refuse to block because acting too aggressively could crash or disable critical systems.
- Products prioritize system stability over acting immediately, sometimes opting to alert instead of block if there is a risk of operational disruption.



3

BUSINESS POLICY EXCEPTIONS

- Organizations often choose to allow certain risky behaviors (like using VPNs, accessing external applications, or permitting social media), for business reasons.
- These policies intentionally weaken prevention, creating windows where threats can bypass defenses.

Blocking that file crippled legitimate operations, leading to widespread system failures. As a result, modern security products perform extensive regression testing before allowing prevention actions, and sometimes opt to alert rather than block if the risk of collateral damage is too high.

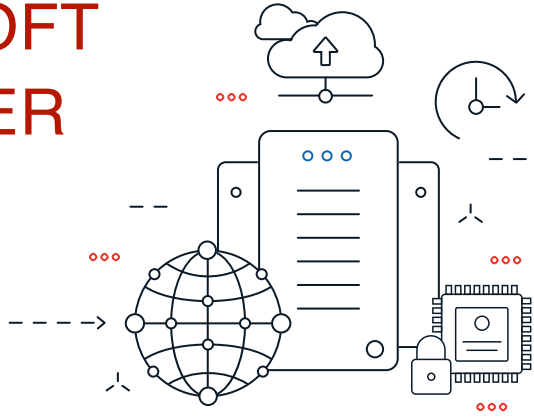
Second, business policies often create intentional exceptions. Organizations may configure security products to allow certain categories of activity, such as the use of VPNs or social media access, in order to support business needs. Unfortunately, these allowances can open pathways for attackers. For instance, a user installing a consumer-grade VPN might inadvertently bypass corporate firewalls and content filters, exposing the device to malicious sites that would otherwise have been blocked.

Security tools operate by interpreting what is happening inside an environment and attempting to classify behaviors as either acceptable or malicious. Prevention technologies do this by inspecting files, analyzing processes, and watching network activity, but their work is fundamentally a matter of interpretation, not certainty. When an antivirus product examines a file, for example, it may first compare the file's SHA-256 hash against a database of known threats. This method is effective for detecting older, cataloged malware, but quickly loses value against newer or modified attacks. To cover this gap, the product will attempt to interpret the file's behavior—sometimes by executing a portion of the code in a controlled environment—to see whether it behaves suspiciously, such as installing a virtual shell or unpacking additional malicious payloads. However, interpretation takes time, and sophisticated attackers understand this. They design their malware to evade or delay detection, using tricks like unusual binary structures, virtual machine detection, or delayed execution, making it harder for prevention tools to react in real time.

Third, prevention technologies sometimes delay action because the threat itself is only partially understood. A suspicious domain may be flagged through DNS analysis, but until patterns of misuse are confirmed, the system may refrain from blocking traffic to that domain. In the meantime, an attacker could exploit the gap.

In short, prevention fails not only because of technical limitations but because of deliberate design tradeoffs. Prevention software is highly effective when facing known threats, but when facing unknown or emerging threats, it is constrained by race conditions, operational risk, and policy exceptions. These limitations are not flaws in design; they are consequences of the complex balancing act between security, usability, and certainty. Understanding these dynamics is essential for appreciating why prevention, no matter how strong, can never be a complete solution on its own.

THE PREVENTION GAP IN PRACTICE: MICROSOFT DEFENDER



Microsoft Defender for Endpoint is one of the most widely deployed security platforms in use today, offering organizations a combination of prevention, detection, and investigation capabilities. Yet despite its maturity and broad adoption, many users struggle to understand when and how Defender fails to prevent a threat. Part of this confusion stems from the way Defender structures its incident and alert information, which introduces layers of complexity that can obscure the real story.

When Defender identifies potentially malicious behavior, it generates an alert. However, this alert is not always a simple, singular record. Often, it aggregates data from multiple users or devices experiencing similar issues within a given time window, such as the past ten minutes. This packaging produces an outer envelope—an incident—containing numerous inner elements such as users involved, devices affected, files encountered, and processes observed. Much like a file directory system, where folders contain subfolders and documents,

Defender's incident model requires the security analyst to navigate through hierarchies of information to truly understand what happened.

Reduced MS Defender Alert


```
{
  "@defender": {
    "alert": {
      "category": "Malware",
      "classification": null,
      "description": "This file exhibits behaviors or
traits of malware. It might do one or more of the
following: \n1. Give a remote attacker access to your
PC. \n2. Download and install other malware. \n3. Record
your keystrokes and the sites you visit. \n4. Send
information about your PC, including user names,
passwords and browsing history, to a remote malicious
hacker. \n5. Use your computer for click-fraud, bitcoin
mining, DDoS attacks and spamming.",
      "detectionSource": "WindowsDefenderAtp",
      "detectorId": "b27a3aa5-ad74-4543-a7db-
f65490d9ff30",
      "determination": null,
      "devices": [
        ... device properties
      ],
      "entities": [
        ... all the objects that were of intrest (there
were 10 of these)
        {
          "aadUserId":
"8888888-8888-8888-8888-88888888888",
          "accountName": "xxxxxxx",
          "detectionStatus": "Detected",
          "deviceId": "88888888888888888888888",
          "domainName": "DESKTOP-88888888",
          "entityType": "Process",
          "fileName": "WebCompanionInstaller.exe",
          "filePath": "C:\\Users\\xxxxxxxxx\\AppData\
\Local\\Temp\\AC8F.tmp\\888888888888",
          "parentProcessFileName": "WcInstaller.exe",
          "parentProcessFilePath": "C:\\Users\\xxxxxx\
\AppData\\Local\\Temp\\AC8F.tmp",
          "processCommandLine":
"WebCompanionInstaller.exe --partner=IS220301  --
version=8.9.0.389 --silent --partner=IS88888888",
```

```
            "remediationStatus": "None",
            "sha1":
"fc1c82f650e7fdecc1070a8a0886500306849818",
            "sha256":
"457b7372266adda3835f47a4113e5f7fc7f2d050dc773b3c3255e91c
79bdf9b0",
            "userPrincipalName": "xxxxxxxxx@example.com",
            "userSid": "S-1-5-2188888888888-888888888-88888
888-8888",
            "verdict": "Suspicious"
          },
        ],
        "investigationState": "UnsupportedAlertType",
        "resolvedTime": null,
        "severity": "Medium",
        "status": "New",
        "threatFamilyName": null,
        "title": "A suspicious file was observed"
      },
      "alertKey": "Multi-stage incident involving Execution &
Discovery on multiple endpoints (310173)",
      "assignedTo": null,
      "classification": "Unknown",
      "comments": [],
      "determination": "NotAvailable",
      "incidentId": 310173,
      "incidentName": "Multi-stage incident involving
Execution & Discovery on multiple endpoints",
      "incidentUri": "https://security.microsoft.com/
incidents/310173?tid=b65483d2-7c0b-4ad9-81ba-3716a433966d",
      "lastUpdateTime": "2025-04-28T02:55:55.8266667Z",
      "redirectIncidentId": null,
      "severity": "Medium",
      "status": "Active",
      "tags": []
    }
}
```


This is just a snippet. That file hash, sha256, though detected by eleven virus companies was missed by CrowdStrike, MS Defender and SentinelOne at the time I wrote this. This is a nasty file, for it is a browser extension/Trojan.

At first glance, an alert describing "malware behavior" may seem reassuring. After all, the system has seen the issue and reported it. Yet a closer inspection often reveals that the situation is far less contained than it appears.

For instance, in a real-world example, Defender detected suspicious activity across several files, processes, and user sessions. The detection status of these artifacts was marked as “Detected,” but their remediation status was listed as “None.” In other words, while Defender had recognized the presence of malware-like activity, it had taken no action to block, quarantine, or neutralize the threat. The system had simply observed and reported, leaving the problem active (status:”Active”) within the environment.

Understanding this distinction requires interpreting several fields inside the Defender alert. The “Action Taken” field may list values like “Blocked,” “Quarantined,” or “Allowed,” but if the action is “Detected” with no accompanying preventive action, the system has failed to stop the threat at the point of contact. Similarly, the “Status” field may differentiate between “Prevented,” “Remediated,” and “Detected,” but again, “Detected” alone does not indicate any protective measure was executed. Perhaps most critically, there is no single, dedicated field within Defender’s schema that explicitly says “Failure to Prevent.” Instead, analysts must infer failure based on combinations of detection status, remediation status, and the absence of system-initiated actions like termination or quarantine.

A clear example of this can be seen in cases where the detection status is “Detected,” the remediation status is “None,” and the investigation state shows “UnsupportedAlertType” or remains unresolved. The entities involved—files, processes, users—may be flagged as “Suspicious,” but no automatic containment or eradication is attempted. In the incident reviewed above, suspicious files such as “asdk.dll” and processes like “5cds3syu.exe” were observed operating within the system without any preventative action being taken by Defender.

Why does this happen? As discussed earlier, prevention technologies encounter several structural challenges. In many cases, latency introduced by the need to interpret unknown or novel threats prevents an immediate block. The product may recognize something is suspicious but require additional time or analysis to determine maliciousness conclusively. Operational risk further complicates matters; acting too aggressively against a suspicious but unconfirmed event could disrupt legitimate system functions, leading to a bias toward caution. Policy settings also play a role: organizations sometimes allow medium-risk activities to proceed for business reasons, inadvertently creating security exposure. Finally, there are circumstances where the prevention engine simply cannot make a definitive decision based on the information available, leading it to flag a behavior without intervening.

These realities illustrate an important truth about Microsoft Defender, and indeed most prevention technologies. Detection does not guarantee prevention. Understanding the difference requires more than simply reading an alert's title; it demands a close reading of multiple fields, a grasp of the underlying logic of the system, and an acceptance that security monitoring must compensate for what prevention does not catch. It is within this nuanced space—the often hidden gap between detection and action—that security teams must build their response capabilities.

WHAT HAPPENS AFTER PREVENTION FAILS

When prevention fails, the pathway to recovery is not determined by the technology itself, but by the user's ability to recognize the failure and respond appropriately. Detection alone is not action. An alert that a file is suspicious or that a behavior matches malware-like patterns does little unless someone translates that information into concrete steps. Without that recognition, the system's inaction remains unnoticed, and the threat persists within the environment.

In the case we examined earlier, the system had identified multiple suspicious files and processes. Within that group, some SHA-256 hashes correlated to known malicious indicators. Yet despite the system's detection, no automatic remediation occurred. The malware remained active, highlighting the critical difference between identifying a problem and resolving it. At this stage, it falls to the security team to take deliberate action.

The first priority is containment. However, containment must be approached with care. It is tempting, especially when malware is detected, to respond aggressively by isolating systems or restricting broad network access. Yet in a mid-sized company, such actions can introduce serious operational disruption. Precision, not panic, must guide the response. The objective should be to remove the malware from the affected system and to validate that the infection has been fully neutralized, without unnecessarily impacting unrelated users or services.

After initial containment and cleansing, attention must turn toward reinforcement. Preventing recurrence requires strengthening the prevention posture itself. This is typically achieved by updating blacklists or prevention rules to recognize and block the identified malware in the future. One common method is to use the SHA-256 file hash to create a direct signature match. Another, often more durable method, is to recognize patterns in the file structure or behavior that are more difficult for an attacker to change. While malware developers can easily alter file hashes or slightly modify binaries, adjusting the deeper structure and behavior of their payloads is a much more complex task, making structural detection a valuable complement to simple hash-based blocking.

This entire process—detection, containment, validation, and reinforcement—is work that the security team must undertake manually when prevention fails. Tools like Microsoft Defender may assist by providing evidence, but they do not automatically repair the situation. Recognizing this is essential for building a resilient security practice. Recovery is not about trusting technology to act on your behalf; it is about understanding the technology's limits and stepping in to close the gaps it inevitably leaves behind.

THE REAL PATH TO SECURITY HEALTH

Security begins with prevention. At its core, the concept of prevention reflects a simple truth: stopping a threat before it takes hold is always better than trying to repair the damage afterward. Building and maintaining a strong prevention layer requires deliberate and ongoing effort—patching systems to remove vulnerabilities, updating antivirus and endpoint detection signatures to recognize new threats, and hardening infrastructure configurations to resist attack. These activities form the bedrock of a secure environment, and without them, no amount of detection or response can truly succeed.

Yet as critical as prevention is, it is not enough on its own. No security system, no matter how comprehensive or well-designed, can promise to stop every threat. Failure is not an anomaly; it is an inevitability. New threats emerge faster than defenses can be updated, trusted processes can be abused, and race conditions—where a system must decide whether to allow or block an action before it has full information—are unavoidable. Recognizing this is not an admission of defeat, but a mark of maturity.

To manage this reality, organizations must build systems not only to prevent attacks but to recognize when prevention has failed and to respond effectively. This requires structured processes, just as formal as those used for patching and system maintenance. In the case of Microsoft Defender, for example, alerts fall into multiple categories, each reflecting different aspects of

potential failure. Some may point to a detection without a block, others to suspicious but unresolved activity, and others to confirmed malicious behavior that has yet to be remediated. Each type demands a different response, and without predefined procedures to address them, organizations are left exposed even after a threat is identified.

The real path to security health, therefore, lies in building prevention, detection, and response into the fabric of operations as formal, managed disciplines. Prevention remains the first and most important step, but it must be complemented by active detection processes that surface failures quickly, and response processes that remediate them before they grow into larger problems. Treating these three pillars—prevention, detection, and response—as operational systems, not as standalone tools, is the only way to create a resilient and secure environment. Security is not the absence of threats; it is the ongoing ability to recognize, manage, and overcome them.

IS THERE A PRODUCT FOR WHAT HAPPENS

When organizations recognize that prevention alone is not sufficient, a natural question follows: is there a tool designed specifically to handle what happens after prevention fails? The answer is yes. Fluency was built precisely to address this overlooked but critical need.

Traditional security products, whether Microsoft Defender, Fortinet, or any number of others, are built primarily around two objectives: preventing attacks and providing general detection telemetry. Their strength lies in catching known threats and enforcing baseline security policies. However, when prevention fails—as it inevitably does—these products often leave a gap. They may generate alerts, but those alerts are not always organized or prioritized in a way that facilitates effective response. In fact, most alerts generated by security products are either confirmation of successful prevention or informational notices that have little actionable value. The result is that critical failures are buried inside volumes of benign data, making it difficult for security teams to know where to focus.

The role of a true SIEM, Security Information and Event Management system, should be to bridge this gap. A SIEM should not simply collect and report on volumes of events; it should intelligently identify when prevention has failed and help guide the organization toward a focused, effective

response. Unfortunately, many SIEM platforms have lost sight of this purpose. Instead of highlighting prevention failures, they overwhelm analysts with sheer event volume, categorizing and counting logs without distinguishing between what is important and what is irrelevant.

Fluency was designed differently. Its core mission is not to tell you what has already been prevented or what is simply happening in the environment. Its purpose is to identify where prevention failed, to surface those failures quickly and clearly, and to provide the enriched context necessary for rapid resolution. Fluency analyzes events not atomically —not as isolated points—but relationally, comparing suspicious events to related activity, performing automatic lookups such as SHA-256 evaluations, and enhancing raw data with actionable intelligence. It allows security teams to move beyond simple detection and toward actual closure of security issues.

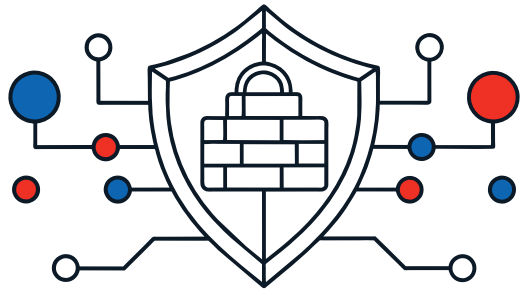
In a situation like the Microsoft Defender case we examined earlier, Fluency would not simply record that a suspicious file was observed. It would correlate that observation with user activity, device health, file behaviors, and external reputation data. It would clearly present the entities involved—the users, devices, and processes—and the objects of interest—the malware artifacts, compromised files, or malicious commands—allowing the security team to scope the problem accurately and act decisively.

This paper has not gone into the full labor that effective response entails, but it is important to understand that without a tool designed to properly detect and frame prevention failures, response cannot happen efficiently. Analysts must be able to recognize the issue, assess its scope, and close the case confidently. Without that

foundation, security operations become a guessing game, chasing alerts without meaningful resolution.

Fluency supports this reality. Microsoft Defender is only one of the many sources it analyzes, but it serves as a perfect example of why a gap-filling technology is necessary. Organizations that focus heavily on prevention often overlook the need for a tool that understands the aftermath, but in truth, without such a tool, the entire security program remains exposed. The gap between prevention and recovery cannot be closed by human effort alone. It requires technology specifically designed to find where prevention ends—and to make that visible, understandable, and solvable.

FREQUENCY OF FAILURE: THE



Coming directly out of the discussion on Fluency, it is timely to confront one of the most misunderstood realities of security operations: the frequency of prevention failure. When discussed casually, failure rates in prevention systems such as Microsoft Defender can seem comfortably low. Measured across large environments, the rate of failure tends to sit between 0.3% and 0.6%, depending on the product and how correlation is applied. On the surface, these numbers suggest an almost negligible risk—an impression that can easily lead to complacency.

However, when you scale those small percentages across the size of a real organization, a very different picture emerges. For an environment of approximately 50,000 users, after applying User and Entity Behavior Analytics (UEBA) correlation—an important capability in Fluency—the result is between fifteen to twenty missed preventions per day. If we adjust this calculation to a more modest organization of 1,000 users, the numbers remain striking.

Even at that smaller scale, an organization will experience roughly twelve missed preventions per month.

Twelve failures a month may not seem overwhelming at first glance, but when framed in the context of what those failures represent—twelve instances of malware or malicious activity that prevention tools did not contain—the risk becomes far more tangible. Each one of those twelve failures represents a point of potential compromise, requiring detection, scoping, and remediation before the damage can spread further.

It is important to recognize that prevention is, in fact, doing an enormous amount of good. In the example above, the environment was generating around 100,000 security-relevant events per day for 50,000 people. The fact that the total number of actionable failures could be reduced to around 600 per month is a testament to the effectiveness of modern prevention techniques. Yet even this high success rate still leaves a meaningful workload for detection and response teams.

This dynamic is not a paradox; it is a known statistical phenomenon. It can be understood through what mathematicians refer to as a Poisson distribution. In simple

terms, a Poisson model explains how rare events, when given enough opportunities, become inevitable. Imagine casting a fishing line into a river once and missing. But if you cast that line a thousand times, you will eventually catch something. Security systems face the same problem. Even with a tiny failure rate per event, the sheer volume of events ensures that failures will accumulate over time.

This reality has significant implications for how security programs must be designed. Failure is not just possible—it is guaranteed. The only question is whether the organization is prepared to detect those failures quickly, respond appropriately, and prevent small compromises from growing into major incidents. Tools like Fluency exist precisely because even a small failure rate, applied across large volumes, creates a measurable and constant operational need. A healthy security posture does not depend on fantasy expectations of perfect prevention; it is built on the sober recognition that vigilance, detection, and response must always stand ready to fill the inevitable gaps.

CONCLUSION: SECURITY MATURITY MEANS ACCEPTING AND MANAGING THE GAP

Security operations are often built on the promise of prevention. For many organizations, success is measured by how many threats are blocked before they cause harm, how often the system holds the line. Prevention is, without question, the first and most critical pillar of a strong security program. But true security maturity is not about believing that prevention will be perfect. It is about recognizing that, inevitably, some threats will slip through, and it is the organization's ability to detect and manage these gaps that determines its real resilience.

Throughout this work, we have examined the nature of the prevention gap—the space between what security products are designed to stop and what they inevitably miss. We have seen how even the best technologies, like Microsoft Defender, struggle with unknown threats, operational risks, and policy exceptions. We have explored how the failure rate, while statistically small, compounds across the vast volumes of daily events to create a consistent, measurable risk even for small and mid-sized companies. Failure is not rare. Failure is baked into the very structure of modern cybersecurity.

Accepting this fact is the first mark of a mature security posture. Planning for it—building the processes,

procedures, and technologies to detect failures and respond rapidly—is the second. No organization can operate safely without response capabilities that match the sophistication of its prevention controls. Detection and response are no longer optional extensions of security; they are mandatory components of survival.

It is here that Fluency becomes indispensable. Fluency was not built to celebrate the success of prevention or to monitor benign activities. It was designed specifically to detect the failures that prevention systems miss—the infections that slip past Microsoft Defender, the behaviors that fall outside Fortinet's protections, the incidents that traditional SIEMs lose in the noise of event volume. Fluency exists to highlight what matters: the small but critical gaps where action must occur.

Without a platform like Fluency, organizations risk being blind to their greatest vulnerabilities. Missed preventions accumulate silently, each one a seed for a potential breach. Relying solely on prevention reporting or raw event monitoring leaves security teams overwhelmed and unable to see where true threats lie. Fluency provides the lens, the prioritization, and the actionable intelligence necessary to close the gap between prevention and true protection.

Security maturity is not a matter of luck or passive hope. It is a disciplined process of prevention, detection, and response working together. It is the acknowledgment that every security control has limits—and the commitment to managing those limits proactively. Organizations that embrace this reality will not only defend themselves more effectively; they will build security programs capable of adapting, responding, and thriving even in the face of inevitable failure.

Without Fluency, the gap remains unseen. With Fluency, it becomes manageable.

Chris Jordan

MIND THE GAP

WHEN PREVENTION FAILS

www.fluencysecurity.com