



runZero

# RESEARCH REPORT

VOLUME 1 • MAY 2024





# Table of Contents

Chapter 1 <b>Introduction</b>	<b>3</b>
Chapter 2 <b>OT &amp; Cloud Impacts on Attack Surfaces</b>	<b>10</b>
Chapter 3 <b>Unusual Assets Are Risky Assets</b>	<b>19</b>
Chapter 4 <b>Some Old Enemies</b>	<b>23</b>
Chapter 5 <b>Emerging Threats</b>	<b>36</b>
Chapter 6 <b>Fingerprints &amp; Snapshots</b>	<b>45</b>
Chapter 7 <b>AI &amp; the Need for Specificity</b>	<b>65</b>
<b>About runZero</b>	<b>69</b>



## Foreword by Rob King

“Plus ça change, plus c’est la même chose” [The more things change, the more they stay the same.]

—Jean-Baptiste Alphonse Karr,  
Les Guêpes, 1849.

The only constant in information security is that this year will be different from last year. Not only will new individual threats emerge, but entirely new classes of threats will make their debut. Some evergreen threats will finally die off, while others will roar back from oblivion. More devices (and more types of devices!) will be connected to networks, and attack surfaces will continue to grow in sophistication and scope.

While this may seem daunting, it’s also very exciting. We do not work in a boring industry, and we get to solve fascinating and complex problems every single day. runZero’s research team exists to discover new and innovative ways to solve these problems and, just as importantly, identify new problems to solve tomorrow.

In this report we share runZero’s observations from our unique perspective as an applied security research team. Our goal is to provide insight into how the security landscape is changing, and recommendations on what you can do to get ahead of these changes.

We hope that you will find our first research report educational and possibly even entertaining. We would appreciate your feedback; if you have suggestions, questions, or comments, please reach out by email via [research@runzero.com](mailto:research@runzero.com).



**Rob King**  
Director of Security Research

# Executive Summary

Tectonic shifts are happening in the cybersecurity industry, brought about by the rapid coalescence of several powerful trends and technological developments that have been years in the making.

First and foremost, vulnerabilities are being exploited at a truly unprecedented pace. And it's working. So much so that the **SEC now requires 8K filings for data breaches**, not to mention the constant flow of news about emerging vulnerabilities and successful compromises across organizations of every size and sector.

While zero day attacks at the network edge have surged, suppliers are struggling to provide timely patches for their products, often leaving customers at the mercy of attackers for days or weeks. In response to the acceleration of exploitation, suppliers are now often releasing indicators of compromise (IOCs) in conjunction with their initial notifications to customers.

Recently, the **xz-utils backdoor** became a stark reminder that supply chains are still under immense attack with catastrophic potential. The incident also catalyzed conversations about what it means to be a responsible consumer of open source products, and what "supplier" means in a shared security model.

Meanwhile, enterprise environments are changing faster than ever. The convergence between operational technology (OT) and information technology (IT) networks is an inevitable conclusion, creating a greenfield of high-value targets for cybercriminals to plunder.

Security programs have matured dramatically over the years, but are still dogged by end-of-life systems, unknown assets, and network segmentation challenges. These time-consuming issues compete for resources with short-term fire drills related to emerging threats and exposures. Defenders continue to juggle scoping, patch management, emergency response, and incident analysis on top of business requirements – all while security budgets shrink.

Our analysis also indicates that large organizations are still struggling with long-standing configuration problems. Remote management services are not in great shape. The trends for outdated TLS stacks, continued use of outdated protocols like SMB v1, and general hygiene issues with the Secure Shell and Remote Desktop Protocols continue unabated, with serious implications for long-term security. The silver lining is that default choices by operating system vendors are making a difference, but not fast enough to reduce the risk to the overall attack surface.

While generative artificial intelligence (Gen AI) and large language models (LLMs) have been touted as the next big thing for security, the reality is more modest. LLMs are helpful in many contexts, but are still prediction engines at heart. As a result, LLMs are limited to helping with the human side of security and struggle to replace expert systems and logic-based decision-making.

## // runZero Research Objective

Amidst all of these dynamics, one thing remains clear: as more and more devices are attached to networks, we need faster ways to focus limited information security resources where they are needed most.

Our objective as a research team is to identify incredibly efficient ways to pinpoint at-risk devices, through both precise fingerprinting and fast outlier analysis, and to quickly get these tools into the hands of our customers and community. This first research report includes insights derived from our data analyses, and also describes how we work as a team with this objective in mind.



# Scope & Methodology

This report is based on a representative, anonymized data sample from the public runZero cloud platform. This sample comprises almost four million assets with nearly fifty million associated, distinct data points, including more than 160 network protocols that have been normalized into 800+ distinct attributes and filtered through more than 17,000 unique fingerprints.



**4M**  
Assets



**50M**  
Data Points



**160+**  
Network Protocols



**800+**  
Distinct Attributes



**17K+**  
Unique Fingerprints

runZero's primary data collection method is the runZero Explorer; a lightweight network point-of-presence that is delivered as software and performs active scans, analyzes traffic passively, and integrates with dozens of applications and services.

runZero Explorers provide a true insider's perspective on global cybersecurity, finding ephemeral devices (phones, watches, cars), devices that normally are less monitored (thermostats, projectors, door locks), and the vast "dark matter" of ad hoc and forgotten networks, alongside the assets already on IT's radar.

# runZero's Unique Perspective

runZero was founded on the principle that applied research makes for better asset discovery, and that better asset discovery is the foundation of modern exposure management. Today, runZero is recognized as the leading edge of Cyber Asset Attack Surface Management (CAASM).

This success is due to the work of the runZero research team: a group of industry veterans with decades of experience in information security. The practical output of their research is the accurate and in-depth identification of assets across cloud, on-premises, and remote environments.

CAASM was born out of the old adage that security teams can't defend what they don't know about. The same goes for assets with unknown attributes like their location, type, and nature. In addition to discovering devices and their associated details, CAASM attempts to methodically uncover the types and severity of exposures impacting those assets, offering defenders a new vantage point to observe the attack surface.

CAASM elevates the discovery and visibility (both to attackers and defenders) of assets to a first-class field under the infosec umbrella, and is now considered foundational and critical components of an organization's information security posture. This dynamic is directly tied to the exponential expansion of attack surfaces and to exposures outpacing defenders' resources.



**HD Moore**  
CEO &  
Co-Founder



**Rob King**  
Director of  
Security  
Research



**Tom Sellers**  
Principal  
Research  
Engineer

The attack surface of an organization is no longer defined by on-premises locations with a known set of managed devices. Today, the attack surface consists of personal mobile phones, smart watches, thermostats in conference rooms, aquarium pumps in the lobby, game consoles in the CEO's living room, and countless other devices, many of which come and go from the network on a regular basis.

**Figure 1: A list of devices with multiple attack surface designations found by runZero. Devices that span attack surfaces can provide entry points for attackers into internal organizational networks.**

Up	Type	Hardware	Outlier	Attack surfaces
●	NAS	D-Link DNS-320L	1	Internal, External
●	Router	Raspberry Pi	3	Internal, External
●	Router	DrayTek Vigor Router	2	Internal, External
●	DVR	Hikvision DVR	1	Internal, External
●	IP Camera	D-Link DCS-5000L_17	1	Internal, External
●	WAP	MikroTik RB951UI-2HnD	1	Internal, External
●	NAS	D-Link DNS-340L	1	Internal, External
●	WAP	Asus RT-N18U	1	Internal, External
●	Router	MikroTik Router	1	Internal, External
●	NAS	D-Link DNS-320L	1	Internal, External
●	NAS	Synology NAS	2	Internal, External
●	NAS	D-Link DNS-320L	1	Internal, External
●	Firewall	Zyxel ZYWALL	2	Internal, External
●	IP Camera	D-Link DCS-2332L	1	Internal, External

## //Oddball Devices in Our Data Set

Our data has uncovered a plethora of unusual connected devices, some of which have sufficient connectivity and services to route traffic, ranging from crockpots to cars.



Crockpot



Vacuum Cleaner



Light Bulb



Light Switch



Thermostat



DVR



Scanner



Voice Assistant



The COVID-19 pandemic resulted in an explosion of the attack surface perimeter. While remote work was previously a perk, suddenly it became the standard for countless organizations. Huge numbers of employees retreated from the office and added their home networks as entry points to the previously gated and walled garden under the CISO's watchful eye.

Once considered an island unto itself, operational technology (OT) and industrial control systems (ICS) have converged with IT and further compounded complexity. The whole world has, with very rare exceptions, settled on Ethernet and the Internet Protocol stack for IT. The vast, chaotic sea of proprietary protocols and competing standards of the OT/ICS world have now joined the fray in earnest, along with all the growing pains that come with it.

Today, the world's living rooms and parking lots have become the CISO's responsibility, as well as its factories and utility grids. In 2024, the US Environmental Protection Agency (EPA) wrote an open letter describing how "disabling cyberattacks" are attacking water and wastewater systems throughout the United States. Not so long ago, these systems were unreachable directly from the wider Internet. Today, many of them are perilously and openly exposed to attackers from around the world.

It is in this world that we, as information security practitioners, now find ourselves. Defining attack surfaces is no longer an academic exercise that can be table-topped once a quarter. As exposures emerge at lightspeed, rapid, real-time discovery and CAASM are more critical than ever before.

Today, the worlds' living rooms and parking lots have become the CISO's responsibility, as well as its factories and utility grids.



# OT & Cloud Impacts on Attack Surfaces

In cosmology, there is the concept of the holographic universe: the idea that a three-dimensional volume of space can be entirely described by the exposed information on its two-dimensional surface.

In the context of an organization's security posture, attack surface is everything; A vulnerability is almost meaningless unless it is exploitable by a bad actor. The trick to determining where the vulnerable rubber meets the exposed road is in identifying what's actually reachable, taking into account security controls, or other defenses in depth.

As described in the previous section, attack surfaces are expanding in multiple ways, becoming more numerous and more specific. Two areas of attack surface growth that merit attention are **operational technology and the cloud**, not only because of their prevalence but also because of the associated risks.

The merging of operational technology and industrial control systems (OT/ICS) devices under the general IT umbrella has created another high-value attack surface. In the past, OT/ICS devices typically had completely separate, dedicated networks. Now they are increasingly attached to enterprise IT networks, making them a valuable and vulnerable part of an organization's attack surface.

Meanwhile, the increasing commoditization and virtualization of infrastructure means that virtually all organizations now have a cloud-based attack surface to protect. Both independently and combined, these shifts have created new footholds for attackers that are worth examining.

# OT & IT Are Converging

Historically, security was of less concern than safety and reliability for the “grade-separated” networks supporting OT/ICS devices. These networks had different dynamics, using industry-specific network protocols and following maintenance schedules that were less frequent than IT systems.

OT equipment is designed for long-term reliability and infrequent changes. The result is that many factory floors, water treatment plants, critical infrastructure, and other industrial processes use equipment that is relatively slow compared to modern PCs. In order to support real-time control requirements, OT equipment often excludes encryption and authentication at the protocol level.

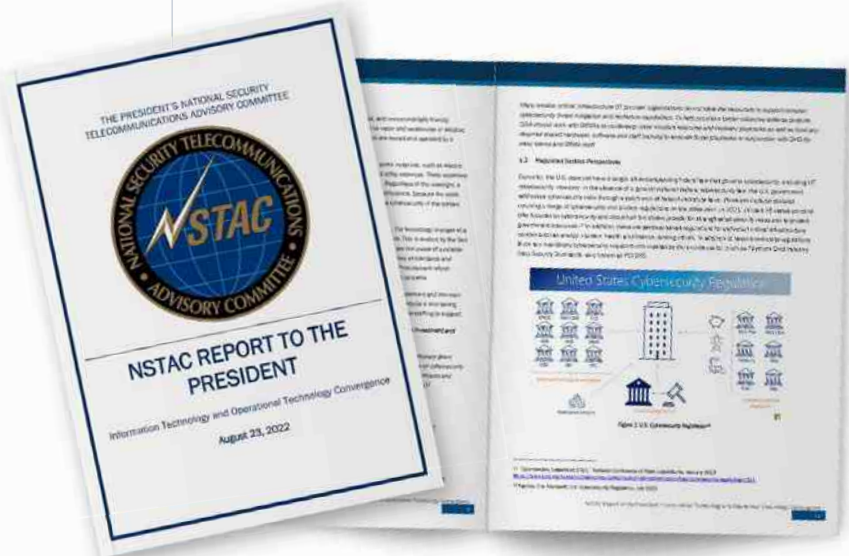
OT systems offered a soft target for malicious actors, but only if they could reach these networks. Until recently, OT was simply not IT’s problem.

Improvements to networking and security technologies have changed this, allowing organizations to link their OT and IT networks (sometimes on purpose, and sometimes not).

Teams that were previously responsible for securing laptops and servers are now also responsible for OT security. With mandates to improve management and monitoring efficiencies, systems that were once in a walled garden are now, at least in theory, reachable from anywhere in the world.

The 2022 report from the President’s National Security Telecommunications Advisory Committee on IT-OT Convergence concludes that we must “accept that IT/OT convergence will be the end state” of IT and OT.

“IT/OT convergence will be the end state”



## OT/ICS AROUND THE WORLD

runZero data confirms that thousands of OT/ICS devices are indeed “reachable from anywhere in the world.” These devices are prime targets for state actors and ransom seekers, as compromising them can result in industrial or utility disruption.

The number of industrial control systems (ICS) directly exposed to the public Internet is terrifying. While the year-over-year increase of exposed devices has finally slowed, the total number continues to climb. **Over 7% of the ICS systems** in this report’s sample data were connected directly to the public Internet, illustrating that organizations are increasingly placing critical control systems on the public Internet.

**Figure 2: A selection of industrial devices detected by runZero on the public Internet.**

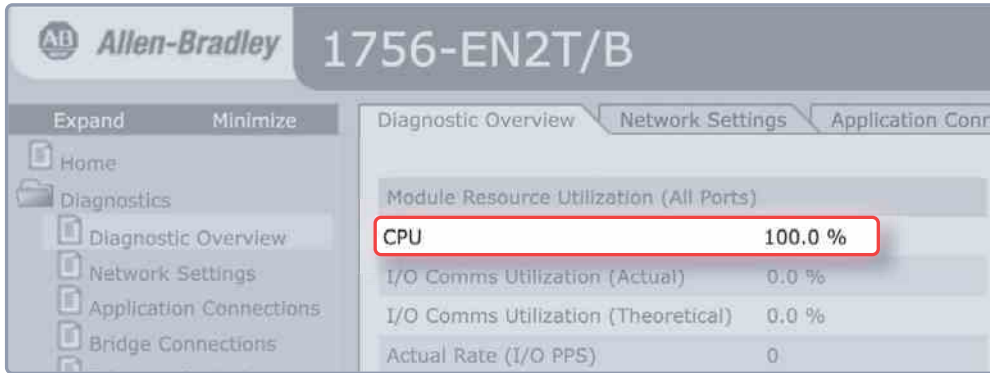
Up	Type	Hardware	Outlier	Attack surfaces	Sources
	PLC	Siemens LOGO! PLC	1	External	
	Ethernet Gateway	Wiesemann & Theis Com-Server Highs...	2	External	
	Data Logger	Schneider Electric ComX	1	External	
	PLC	Allen-Bradley 1766-L32BXBA	2	External	
	Ethernet Gateway	Schneider Electric PowerLogic EGX1...	1	External	
	Power Device	SMA Solar Technology AG Sunny WebB...	1	External	
	PLC	Rockwell Automation PLC	1	External	
	BACnet	Schneider Electric BACnet	1	External	

## OT SCANNING MOVES FROM PASSIVE TO SOMETIMES ACTIVE

OT devices often run industry-specific software on hardware with limited computing power. These constraints, combined with the long-term nature of OT deployments, result in an environment that does not respond well to unexpected or excessive network traffic.

Stories abound of naive security consultants accidentally shutting down a factory floor with vulnerability scans. As a result, OT engineers have developed a healthy skepticism for any asset inventory process that sends packets on the network and instead opted for vendor-specific tools and passive network monitoring. Passive monitoring works by siphoning network traffic to an out-of-band processing system that identifies devices and unexpected behavior, without creating any new communication on the network.

**Figure 3: An Allen-Bradley industrial PLC indicating 100% CPU utilization due to the device receiving a high rate of packets from an active scan NOT conducted by runZero.**

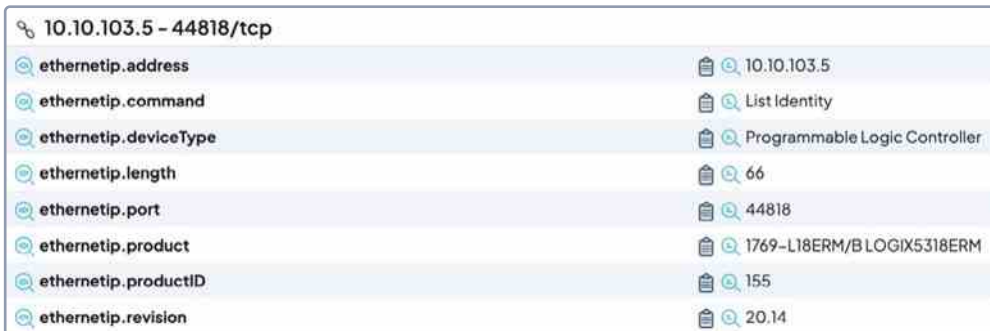


While passive discovery is almost entirely safe, it is also limited. By definition, passive discovery can only see the traffic that is sent, and if a device is quiet or does not send any identifying information across the network, the device may be invisible.

Passive deployments are also challenging at scale, since it's not always possible to obtain a full copy of network traffic at every site, and much of the communication may occur between OT systems and never leave the deepest level of the network.

Active scanning is faster, more accurate, and less expensive to deploy, but most scanning tools are not appropriate or safe to use in OT environments. Active scanning must be performed with extreme care. Large amounts of traffic, or traffic that is not typically seen by OT devices, can cause communication disruptions and even impact safety systems.

**Figure 4: A partial screenshot of an OT device detected by a runZero active scan.**



## SAFE ACTIVE SCANS

runZero enables safe scans of fragile systems through a unique approach to active discovery. This approach adheres to three fundamental principles.

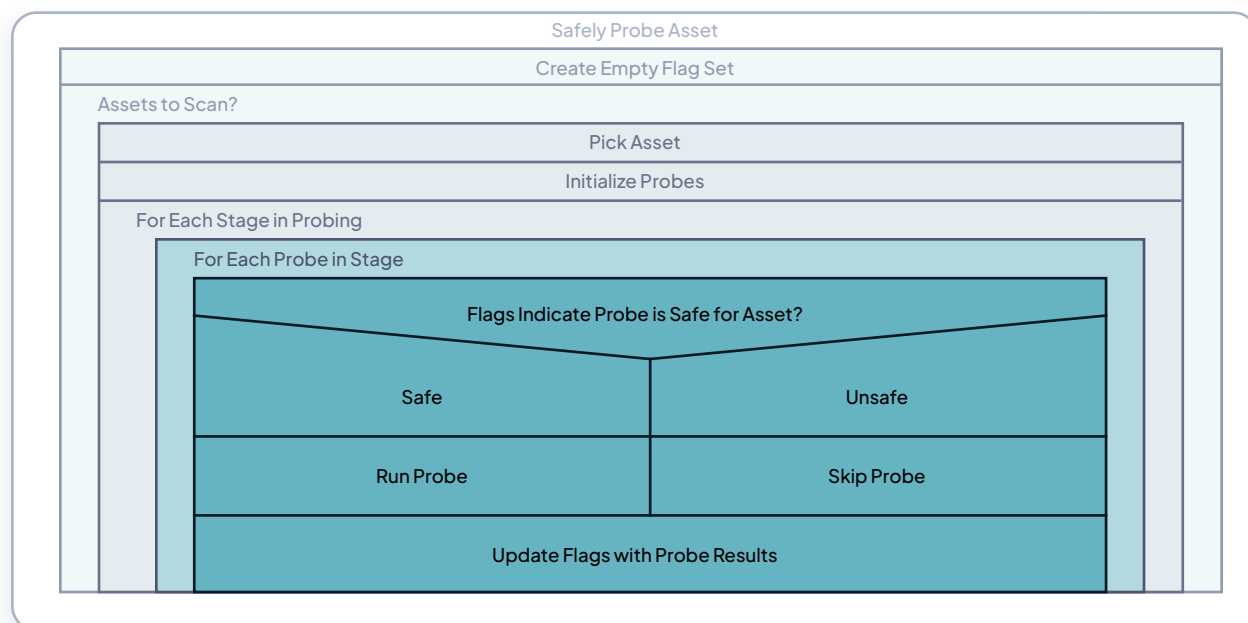
- Send as little traffic as possible
- Only send traffic that the device expects to see
- Incrementally discover each asset to avoid methods that may be unsafe for a specific device

runZero supports tuning of traffic rates at the per-host level as well as globally across the entire task. runZero's active scans can be configured to send as little as one packet per second to any specific endpoint, while still quickly completing scans of a large environment at a reasonable global packet rate.

runZero is careful to send only valid traffic to discovered services and specifically avoids any communication over OT protocols that could disrupt the device. This logic is adaptive, and runZero's active scans are customized per target through a policy of progressive enhancement.

runZero's progress enhancement is built on a series of staged "probes." These probes query specific protocols and applications and use the returned information to adapt the next phase of the scan for that target. The earliest probes are safest for any class of device and include ARP requests, ICMP echo requests, and some UDP discovery methods. These early probes determine the constraints for later stages of discovery, including enumeration of HTTP services and application-specific requests. The following diagram describes how this logic is applied.

**Figure 5: A high-level overview of the "progressive enhancement" probing process.**



Lastly, runZero's active scans also take into account shared resources within the network path. Active scans will treat all broadcast traffic as a single global host and apply the per-host rate limit to these requests. Scans that traverse layer 3 devices also actively reset the state within session-aware middle devices using a patent-pending algorithm. This combination allows runZero's active scans to safely detect fragile devices and reduce the impact on in-path network devices as well as CPU-constrained systems within the same broadcast domain.

For those environments where active scanning is inappropriate or unavailable, runZero also supports comprehensive passive discovery through a novel **traffic sampling** mechanism. This sampling procedure applies runZero's deep asset discovery logic to observed network traffic, which produces similar results to runZero's active scanner in terms of depth and detail.

## The Cloud Is Someone Else's Attack Surface

The commoditization of computing power, massive advancements in virtualization, and fast network connectivity have led to just about any form of software, hardware, or infrastructure being offered "as a service" to customers. Where companies used to run their own data centers or rent rack units in someone else's, they can now rent fractions of a real CPU or pay for bare metal hardware on a per-minute basis.

Cloud migrations are often framed as flipping a switch, but the reality is that these efforts can take years and often result in a long-term hybrid approach that increases attack surface complexity. The result is more systems to worry about, more connectivity between systems, and greater exposure overall.

## CLOUD MIGRATIONS

### // Migration Realities

Over the last five years, runZero has observed and assisted with dozens of cloud migration projects. These projects often take longer than planned and result in more assets to manage at completion.

A common approach to cloud migrations is to enumerate the on-premises environment and then rebuild that environment virtually within the cloud provider. runZero helps customers with this effort by providing the baseline inventory of the on-premises data center and making it easy to compare this with the new cloud environment. During this process, organizations may end up with more than twice as many assets, since the migration process itself often requires additional infrastructure.

The migration process can be tricky, with a gradual approach requiring connectivity between the old and new environments. Shared resources such as databases, identity services, and file servers tend to be the most difficult pieces to migrate; however, they are also the most sensitive components of the environment.

The result is that many cloud environments still have direct connectivity back to the on-premises networks (and vice-versa). A compromised cloud system is often just as, if not more, catastrophic to an organization's security situation as a compromised on-premises system.

Ultimately, the lengthy migration process can lead to increased asset exposure in the short-term due to implied bidirectional trust between the old and new environments.



## NEW EXPOSURES

Cloud providers assume many of the challenges with data center management; failures at the power, network, storage, and hardware level now become the provider's problem, but new challenges arise to take their place including unique risks that require a different set of skills to adequately address.

Cloud-hosted systems are Internet-connected by definition. While it's possible to run isolated groups of systems in a cloud environment, cloud defaults favor extensive connectivity and unfiltered egress. Although cloud providers offer many security controls, only some of these are enabled by default, and they function differently than on-premises solutions.

Cloud-hosted systems are also vulnerable to classes of attacks that are only significant in a shared computing environment. CPU-specific vulnerabilities like **Meltdown**, **Spectre**, and **Spectre v2** force cloud operators to choose between performance and security. The mitigations in place for these vulnerabilities are often bypassed. For example, the recently-disclosed **CVE-2024-2201** allows for Spectre-style data stealing attacks on modern processors, a concern in shared-hosting cloud environments.

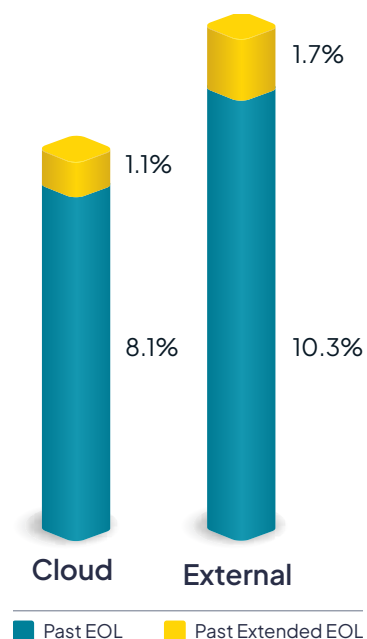
Additionally, the ease of spinning up new virtual servers means that cloud-based inventory is now constantly in flux, often with many stale systems left in unknown states. Keeping up with dozens (or even thousands) of cloud accounts and knowing who is responsible for them becomes a problem on its own.

We analyzed systems where runZero detected end-of-life operating systems (OSs), and found that the proportions of systems running unsupported OSs are roughly the same across the cloud and external attack surfaces. This implies that the ease of upgrading cloud systems may not be as great as advertised.

**Figure 6: CPU-specific vulnerabilities targeting cloud operators.**



**Figure 7: Comparison of end-of-life operating system distribution between cloud and external attack surfaces.**



## HYBRID IS FOREVER

Cloud infrastructure is here to stay, but so is on-premises computing. Any organization with a physical presence – whether retail, fast food, healthcare, or manufacturing – will require on-premises equipment and supporting infrastructure.

Cloud services excel at providing highly available central management, but a medical clinic can't stop treating patients just because their Internet connection is temporarily offline. A hybrid model requires faster connectivity and increasingly powerful equipment to securely link on-premises and cloud environments.

Even in more simplistic environments, cloud migrations leave behind networking devices, physical security systems, printers, and file servers. All of that equipment will most likely be linked to cloud environments, whether through a VPN or over the public Internet.

While cloud migrations can help organizations modernize, these environments still require equally comprehensive asset visibility and exposure management with on-premises infrastructure.



# Unusual Assets Are Risky Assets

The beginning of many great projects start with a researcher saying “that’s odd.”

One of these moments led to the runZero concept of the “outlier score,” a single, simple numeric value that quantifies how “different” an asset is compared to its neighbors and, importantly, the discovery of how that difference corresponds to the risk associated with a given asset.

Identifying risky assets is fundamental to successful exposure management programs, but this process can be challenging due to the quantity and sources of data. In this chapter, we will explore how runZero outlier scores can be used to quickly identify risky assets, even in cases where no vulnerability management data is available.

## // Identifying Outliers

Our research shows that the outlier score, defined as how unique an asset is within the context of its neighbors, strongly correlates with the risk ranking sourced from third-party vulnerability management data, providing organizations another valuable method to pinpoint potential exposures.

Risk



critical

Outlier score

500

# Calculating Outlier Scores

runZero ingests data from third-party integrations such as vulnerability scanners, device management systems, and security analysis tools, and then combines these data points with our own analysis. This produces a unified numeric **risk rank** for an asset. In general, the “riskier” an asset appears to be, the higher its rank becomes on the risk scale.

runZero looks for assets that differ significantly from their peers, using multiple dimensions and points of comparison. The more an asset deviates from the site’s “baseline,” the greater its outlier status from runZero’s perspective. This is quantified in a single number, known as the **outlier score**: the more unusual a device is in its context from the baseline, the higher its outlier metric will be. The outlier metric starts at zero, but has no practical upper bound (though values >600 are fairly rare).

Identifying outliers breaks through the noise by highlighting assets that merit further investigation by security teams. Often, sites will have large numbers of very similar assets – a server farm of systems running Linux and some routers and switches, or an office with a large number of Windows PCs and printers. In those instances, a small number of, or even a single instance of, unusual devices may indicate that there is an asset that has escaped notice and attention by staff.

## // Research Note

Note that for sites with no “common baseline” of assets, no outlier scores are computed. If they were, everything would be an outlier!

# Outliers Are Riskier, on Average

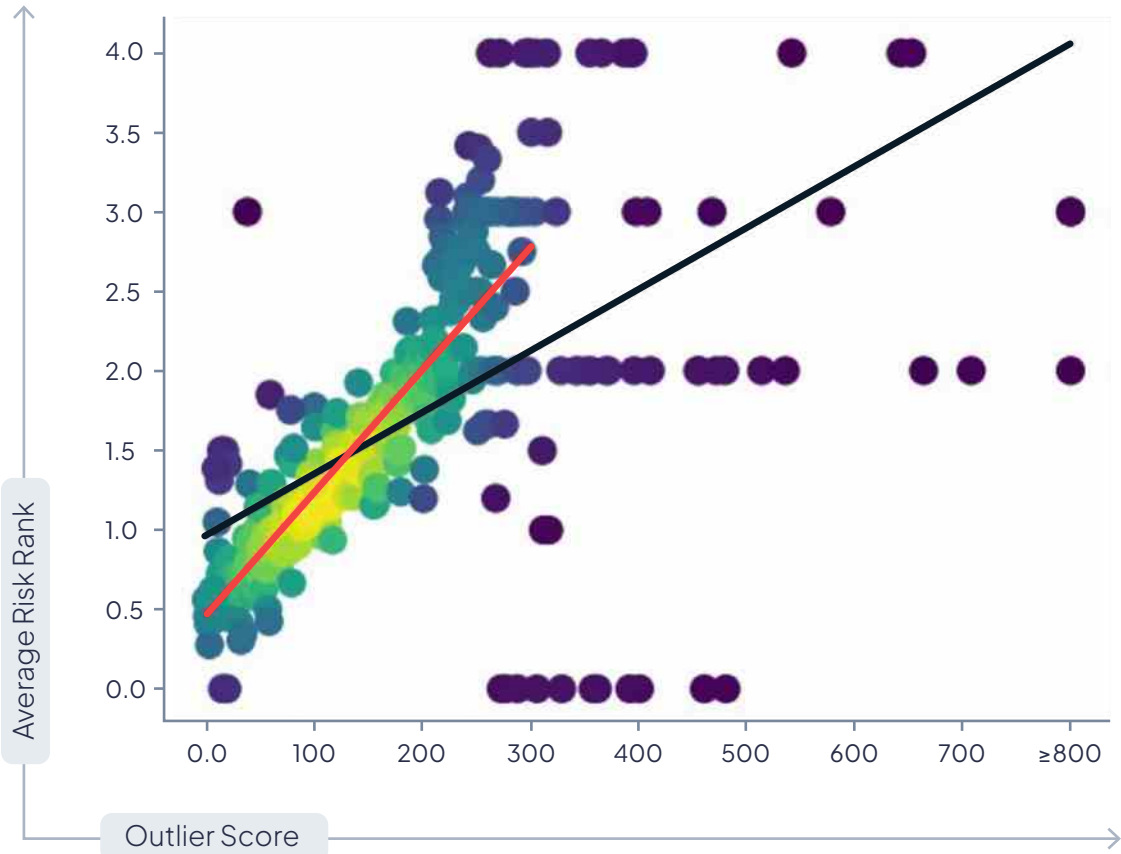
runZero observed that outlier scores are strongly predictive of asset risk. An analysis of 680,000 assets across sites in a variety of organizations and settings revealed that this correlation is quite strong.

Figure 8 illustrates the correlation between the average risk rank of an asset with a given outlier score; color intensity indicates overlapping data points. Lines of best fit are provided for all data points (black), and those with an outlier score  $\leq 300$ ; higher outlier scores indicate that an asset is more unusual.

By definition, most assets have a relatively low outlier score, but note that there is a particularly strong correlation between outlier score and risk rank. It is especially notable that very few devices have an outlier score  $\geq 300$  without also having a risk rank  $\geq 2.0$ .

This correlation is particularly notable for its predictive power. In general, an asset with an outlier score  $\geq 250$  has a 78% chance of having a risk rank  $\geq 2.0$ . Importantly, the opposite also generally holds true: an asset with an outlier score of  $\leq 250$  has a 69% chance of having a risk rank  $\leq 2.0$ .

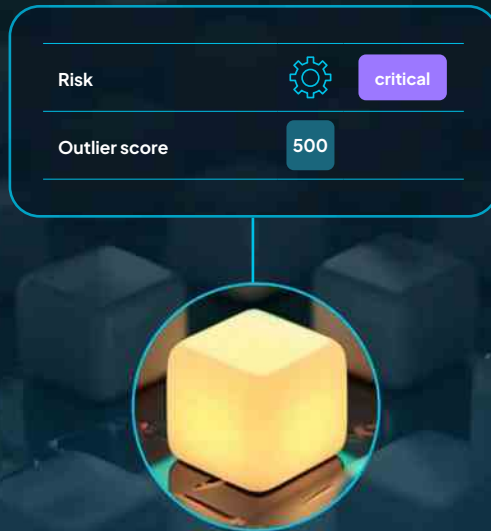
Figure 8: Average risk rank versus runZero outlier score.



# Guidance

An unusual asset may be riskier than its peers, but that doesn't guarantee that it will be noticeable. The outlier score can often reveal when assets differ from the baseline in ways that would not necessarily be apparent to staff: for example, an unusual round-trip time on TCP connections, a slightly different set of services running, and so on.

runZero's research shows that these assets, even if just slightly unusual, are often significantly riskier than others. The outlier score gives security practitioners a powerful tool to find riskier assets – even when no one else might notice.





# Some Old Enemies

Make new friends, but keep the old:  
one is silver, the other gold.

Despite enormous advances within information technology, security practitioners are still plagued by common problems. Advances in secure-by-default designs, zero-trust architectures, and overall security awareness all help, but organizations still struggle with end-of-life assets, network dark matter, and segmentation challenges. These problems are difficult to solve and they often exist outside of defenders' areas of control. Most importantly, from the attacker's perspective, they still provide easy footholds into an environment.

## End-of-Life Is Not the End

All of the system hardening and security patches in the world cannot protect a system that is not updated to use those features. System vendors generally provide patches and updates for a limited timespan. At that point, end users must invest in an upgrade to a newer version of the system or fend for themselves and hope for the best with an end-of-life (EOL), outdated asset lurking on the attack surface.

EOLed systems often stick around for years, mostly forgotten but still part of an organization's infrastructure and, therefore, its attack surface. New vulnerabilities are still discovered and exploited in these outdated systems as the April 2024 [D-Link NAS](#) issue illustrated. Despite the known exposure, being EOL means that fixes will not be forthcoming.

While this may seem like an academic exercise, EOLed systems are surprisingly common. Our findings show many still-active EOLed operating systems in various environments.

## OPERATING SYSTEM END-OF-LIFE

Operating systems typically have multiple phases of vendor support, referred to as a support lifecycle. The duration of the lifecycle and services provided in various stages vary from vendor to vendor, usually tapering off with fewer updates and patches in later stages.

The two phases we are most concerned with are:

**Mainstream support:** during which vendors release patches that may add new features, fix bugs, or mitigate security vulnerabilities.

**Extended support:** during which only critical bugs and vulnerabilities are addressed.

While some vendors' terminology and phases may slightly differ, generally speaking, most support lifecycles can be broadly mapped to these two phases.

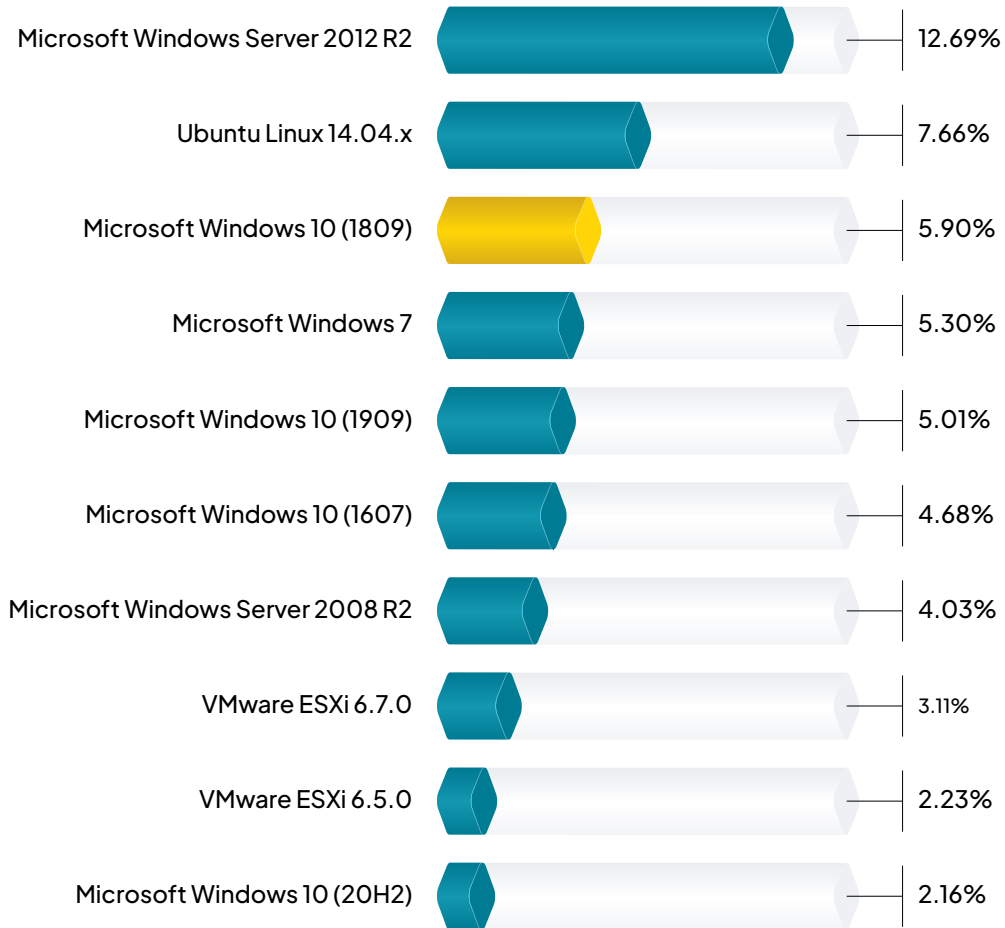
When a vendor stops providing upgrades for non-critical issues, the product is considered in an "End-of-Life" (EOL) status. There may be an additional period known as "Extended-End-of-Life" (EEOL) during which the vendor continues to provide updates for critical issues. EOL and EEOL can happen concurrently or separately depending on the system and the vendor. Most importantly, after EOL, systems no longer receive critical updates or security patches, and thus become much greater risks to keep around.

But around they are! Systems have a long tail: if they still work, replacing them with a supported alternative may be more trouble than it's worth. In some cases, the responsible staff can't or won't; in others, the system may host critical functions that are not supported on newer systems. Uptime guarantees and financial considerations may also play a role.



When we look at our sample data for operating systems that are past their extended EOL dates, we see that the majority are running some version of Microsoft Windows:

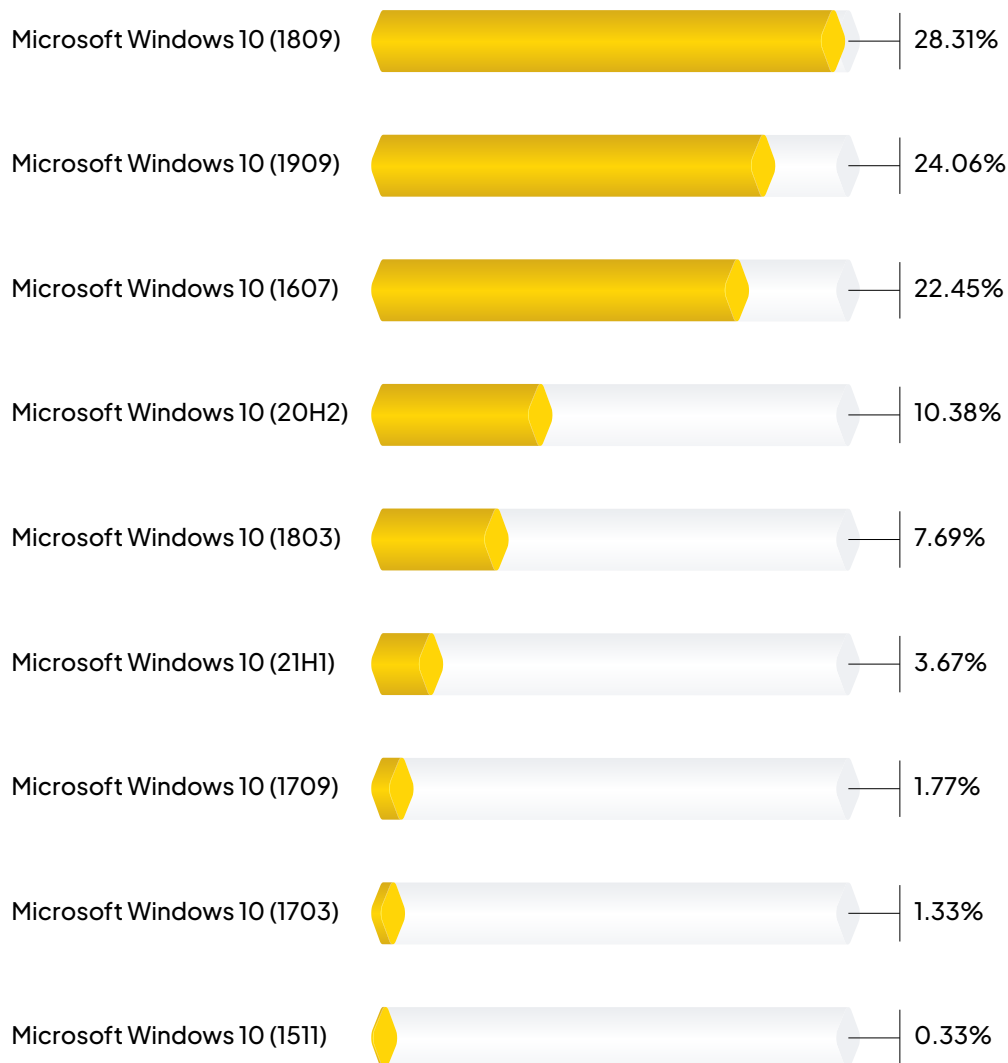
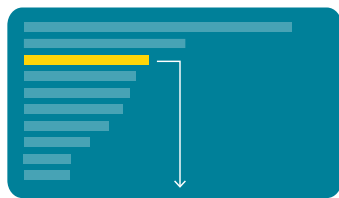
**Figure 9: Top OS past extended EOL.**



The presence of Windows Server 2012 R2 isn't very surprising; it reached extended EOL only very recently, in October of 2023. While unfortunate, it's not unusual for server migrations to drag on past EOL dates due to logistical and compatibility concerns.

The second major group is composed of various Windows 10 releases. Windows 10 was originally released in July of 2015. Microsoft has generally released two major updates for it every year since. Typically, updates released in the first half of the year are supported for 18 months and those released in the second half are supported for 30 months. There are some variations on this theme, with Long-Term Servicing Channel (LTSC) editions, for example, having longer lifespans.

**Figure 10: Windows 10 past extended EOL.**

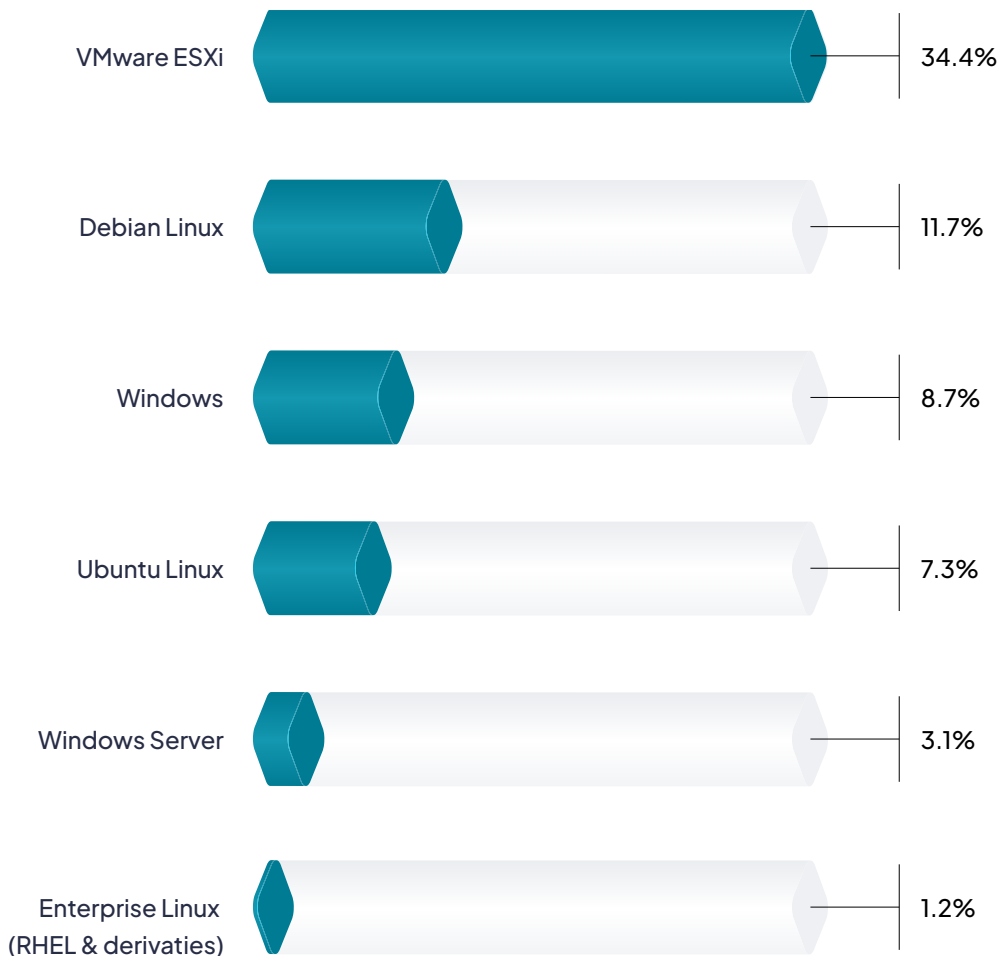


## EXPOSED SYSTEMS PAST EXTENDED EOL

While operating systems outside of their extended lifespans are always worth looking into, those with exposure to an external attack surface are particularly worrisome. Of all systems exposed to an external attack surface and for which EOL data was available, **7.9% were past their extended EOL dates**. That means that roughly 8% of all devices exposed to external attackers are probably not receiving security updates.

For server operating systems specifically, approximately 6% were past their extended EOL dates, with the individual percentages varying across operating systems. Of particular note is VMware ESXi, with over a third of exposed systems being past their extended EOL date.

**Figure 11: Server operating systems with external attack surface exposure, past extended EOL.**



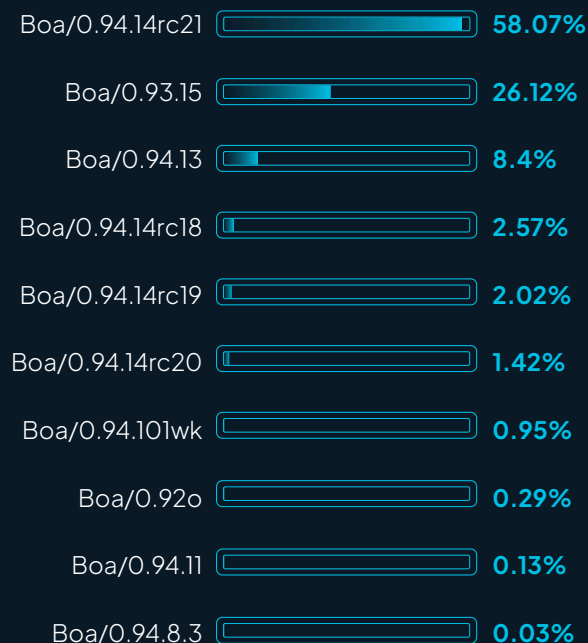
## CASE STUDY: THE BOA WEB SERVER

The **Boa webserver** is an open source web server designed to have low resource requirements for users and to be compatible with embedded applications. The last official release of the Boa webserver, version 0.94.14rc21, was in February of 2005. For comparison, the Colts have won a Super Bowl more recently than the latest release of the Boa web server, and the Colts haven't won a Super Bowl since 2007!

There are known vulnerabilities in Boa that have been exploited in critical infrastructure in the past. For example, in November 2022, Microsoft **disclosed** that Boa web servers in Internet-of-Things (IoT) devices were a common attack vector against power grids in India.

While it is relatively easy for an administrator to determine if a server is running Boa, it is much harder to detect in an embedded device. Boa is common in embedded devices like security cameras and IP phones that are widely deployed in enterprise networks. Therefore, curating an accurate inventory of an organization's embedded devices, not just servers, that are running Boa is critical for protecting these networks.

**Figure 12: Boa web server version distribution in runZero data.**



**Figure 13: Device types still running Boa in sample runZero data.**

Network-Attached Camera	92.3%
Media & Telephony Devices	5.5%
Environmental Control Devices	0.9%
Network Devices	0.9%
Industrial Control Devices	0.3%

# Dark Matter: IoT & Embedded Devices

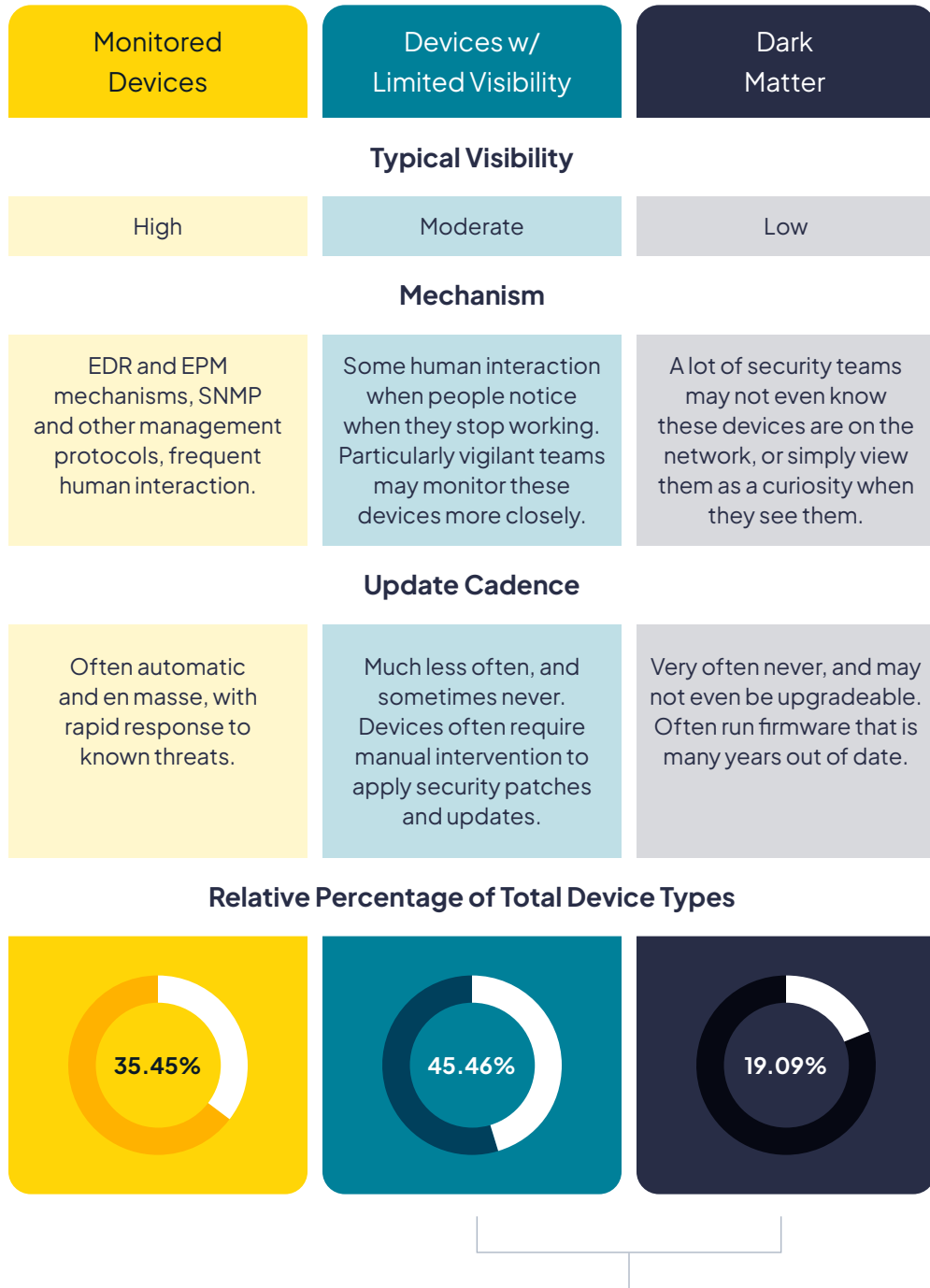
The level of attention, monitoring, and updates that network-connected devices receive can be divided into a three-tier hierarchy, a “hierarchy of visibility” as it were.

At the top level, there are devices that humans interact with directly, or form part of a production system: our laptops, desktops, servers, routers, and switches. They tend to have high visibility to the information security team via mechanisms like Simple Network Management Protocol (SNMP) and endpoint management (EPM) software. These systems usually get frequent, managed updates that are often installed automatically and en masse.

The middle tier consists of “limited visibility” devices present in every office: smart TVs and projectors, media devices like Rokus and Apple TVs, wireless access points, and printers. These devices often support updates over the network but may not receive frequent updates and may require manual intervention to apply them. How often do we think about updating the firmware on our venerable Brother printers?

And last but not least, the dark matter of networks makes up the bottom tier. Much like dark matter in cosmology, these devices are present on the network and their influence can be felt, but they are mostly invisible to IT and management tools. These are things like thermostats, smart plugs and lights, **aquarium pumps**, refrigerators, sprinkler systems, physical access control systems, and so on. These devices often fade into the background and can go relatively unnoticed for years. Updates are likely infrequent or nonexistent, and may require manual intervention if they can even be applied at all.

**Figure 14: Device types by visibility.**



Devices in the last two tiers often outnumber the “visible” devices, sometimes significantly. Analysis of the runZero cloud data for physical assets (excluding virtual machines) indicates that limited visibility devices make up a whopping **45.46%** of discovered devices, with true dark matter devices making up a further **19.09%**.

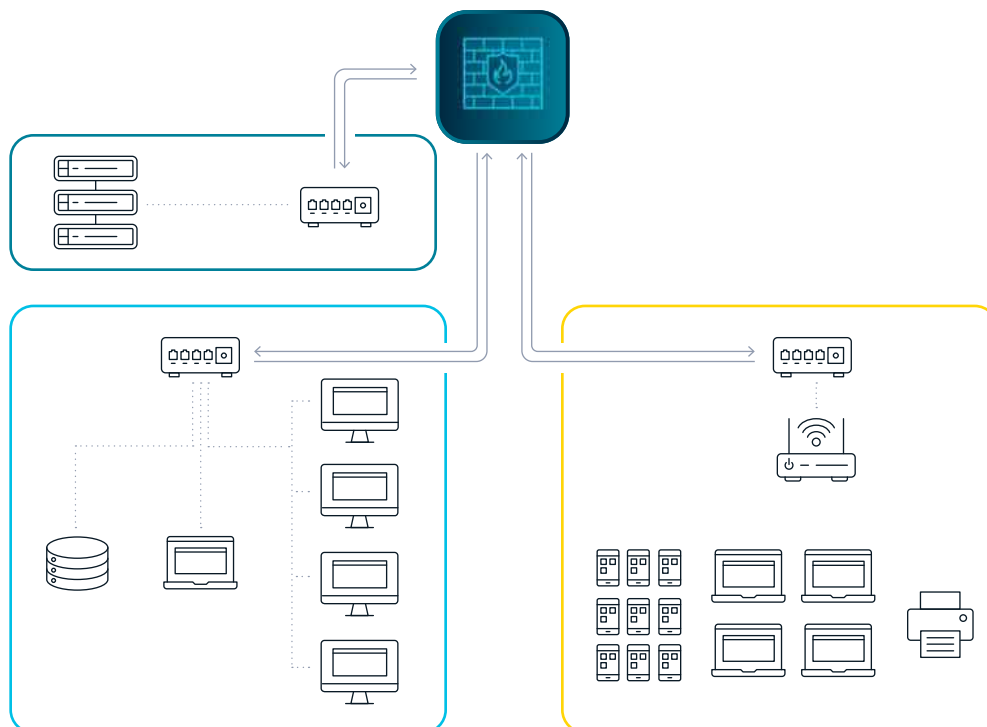
# The Decay of Segmentation

The premise behind network segmentation is that security can be improved by preventing communication between systems with different trust levels and business functions. A company may offer a wireless guest network to office visitors, but does not want those visitors to be able to talk to critical file servers or security equipment.

Segmentation is the most popular approach to securing unmanaged devices; if we are unable to enforce policies on the device itself, placing that device on a separate network that has little access to anything else can reduce the risk of a compromise. With the massive increase of “smart” IoT devices, even ISP-furnished residential routers now offer segmentation features.

Segmentation is a widely-accepted approach to improving network security, to the point where overlooking network segmentation can violate the requirements of common industry requirements and best practices, such as PCI DSS.

**Figure 15: A simple example of network segmentation.**



Segmentation as a goal is great, but it's prone to failure, and often in ways that are not obvious to the team responsible for its implementation. Segmentation assumes that systems are organized into groups based on their business function and that those systems are prevented from communicating with other groups. Cracks appear quickly as the number of systems in the segment grows, primarily for two reasons:



Each additional system in a network segment has the ability to weaken the security of the whole segment. This happens when a system is added that exposes additional network services or that accepts a different form of authorization relative to its peers. Any compromise of a system in that segment can provide a foothold for additional attacks, many of which are only possible when the attacker is on the same local network. The wider the variety of services and authentication sources available, the greater the chance of a successful attack, and this exposure scales with every addition.



Modern equipment is complex and it is rare for any system to have a single network interface. Nearly every device we interact with has more than one way to communicate. The phones in our pockets may support a dozen concurrent network interfaces at once (wireless, Bluetooth, 4G/5G, AWDL, NFC, and more). A modern laptop ships with a half-dozen interfaces out of the box. The humble network printer arrives with wireless, Bluetooth, and Ethernet enabled by default. Segmentation assumes that placing a device on a network limits its access, but that isn't true when every device has multiple network interfaces, and these interfaces allow an attacker to hop between physical connections using wireless protocols.



## CASE STUDY: THE OFFICE PRINTER

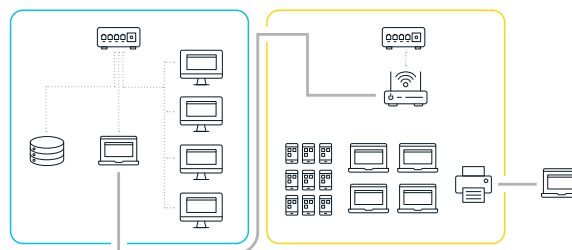
The humble office printer is a great example of the challenges with network segmentation. A typical all-in-one (or multi-function center, aka MFC) printer arrives with Ethernet, Bluetooth, and wireless networks configured out of the box. The printer's wireless interface is exposed as an open access point, allowing anyone within physical range of the device to connect and communicate with the printer. For many reasons, printers are often connected directly to wired Ethernet and the wireless interfaces are still left enabled.

From a security perspective, any device with multiple networks (known as multi-homed devices) can introduce risk, but how risky is this printer example? If an attacker is able to compromise the printer and relay traffic, certainly that is bad, and while it's been demonstrated repeatedly (including attacks via the Fax modem), let's assume that future printer firmware is more secure, and opportunities to directly compromise the printer are less common.

Unfortunately, printers often ship with another tricky feature, and one that isn't possible to disable: IP forwarding. Many printers act like network routers and offer no way to configure this behavior outside of disabling all but one network interface.

IP forwarding allows an attacker connected to one side of a device to route packets through to a network on the other side. This doesn't always mean that network address translation (NAT) is enabled, but even one-way packet delivery into a target network can be disastrous, as source-address spoofing can be used to force target devices to reply to an Internet-facing public IP, providing a two-way communication channel into what should be an isolated network.

**Figure 16: An example of a segmented network, with several devices potentially breaking that segmentation.**



### // Research Note

Even single-interface devices with IP forwarding enabled can be abused to force a device to repeat a message from its own MAC address and network.

Printers are not the only type of device that forwards IP traffic between network interfaces by default; runZero tests for IP forwarding during active scans, and has identified this behavior across IP telephones, network storage appliances, media servers, network cameras, DVRs, battery backup units, smart TVs, video game consoles, and even smart light bulbs. Even industrial automation equipment from HVAC controls to programmable logic controllers (PLCs) inexplicably enable IP forwarding.

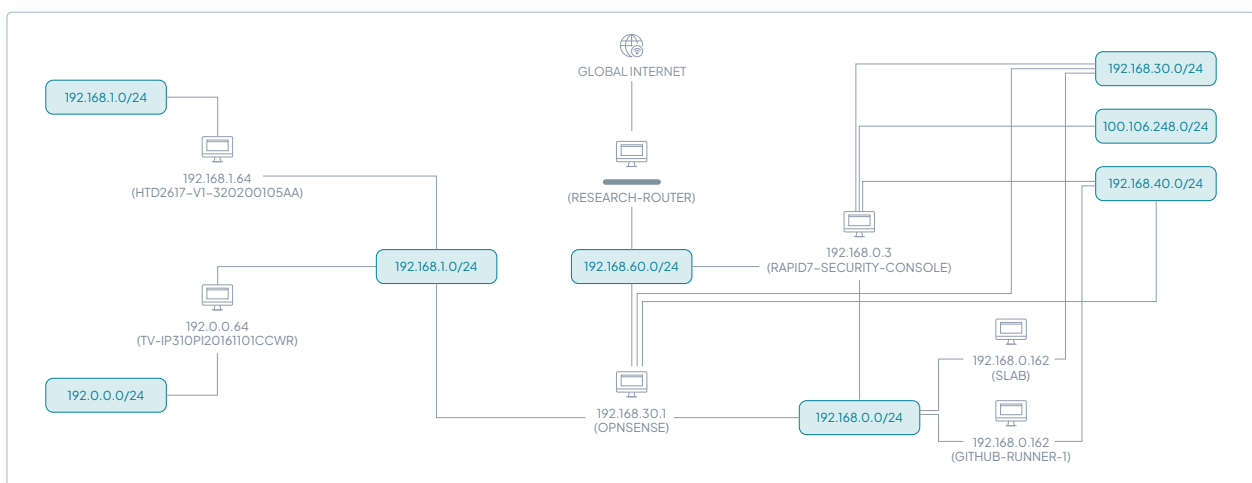
Why does this happen? Oftentimes these devices have virtual network interfaces that are only visible to the firmware itself. To communicate across these virtual interfaces, IP forwarding must be enabled, and no firewall rules were added to prevent the forwarding capability from accepting packets on external interfaces.

This behavior can also surface on servers and workstations. For example, if a developer is using containers for development on their laptop, the container environment often creates virtual network interfaces, and enables IP forwarding for communication across the interfaces. Just like the case of the network printer, if that laptop is connected to both a wired and wireless network, the IP forwarding feature effectively turns their laptop into a router between these segments, since no firewall rule prevents it.

Segmentation is still one of the best tools we have to improve security, but its limitations are becoming more obvious with modern equipment.

In 2024, nearly every device can be a router.

Figure 17: A network diagram showing unexpected network bridging points.



**Figure 18: Unusual devices with IP forwarding capabilities. Box sizes indicate relative frequency in our dataset.**





# Emerging Threats

## Change is the only constant.

As new technologies come into play and older technologies evolve, novel vulnerabilities can emerge. The runZero research team constantly seeks to uncover previously unknown threats, to track emerging threats, and to identify threats that may become consequential in the near future.

Over the last year, the runZero research team has noted some significant trends. Exploitation of emerging threats is happening at an unprecedented pace, shifting the dynamics around zero day vulnerabilities. We've also observed a massive uptick in attacks on secure gateway devices and an increasing number of zero day vulnerabilities exploiting critical products, especially border gateway devices, such as Ivanti Connect Secure systems. And finally, supply chain attacks are more sophisticated and foreboding than ever before.

## Zero Days Are -1 Days

The term zero day (sometimes spelled 0day) describes a vulnerability that is exploited "zero days" after a vendor knew about it - in other words, vulnerabilities that are known to and exploitable by attackers before they are known to vendors.

Zero day vulnerabilities are critical threats, because they essentially place all of the power in the hands of attackers. They are one reason why defense-in-depth is so important: the only way to stop a zero day is to make sure it never reaches its target.

### Notable Emerging Threats



Once a zero day vulnerability is discovered, the clock starts ticking. How quickly can a vendor release a patch for the vulnerability? How many systems will be compromised before the patches can be applied? Will every system get patched? How can we know?

It is absolutely critical that potentially vulnerable systems be located as quickly as possible when a zero day is discovered so that they can be patched (if a patch is available) or removed from potential attack paths (if not).

At runZero, our Rapid Response procedure is invoked when a new zero day is discovered, with the goal of creating mechanisms that empower customers to quickly find and protect vulnerable assets in their inventory. By leveraging data that has already been captured, this can be accomplished without rescanning, providing immediate visibility for existing assets in addition to finding new potentially vulnerable assets going forward.

## // Research Note

In the first four months of 2024, runZero published 23 Rapid Responses covering 60+ distinct vulnerabilities. Notably, more than half were vulnerabilities that were being actively exploited in the wild.

## CASE STUDY: CISCO IOS-XE



The Cisco IOS-XE Web UI vulnerability and subsequent mass-compromise made information security headlines in late 2023. Cisco IOS-XE is a Linux-based operating system for Cisco's high-capacity routers and switches.

In October of 2023, Cisco reported two vulnerabilities in IOS-XE. The first allowed an attacker to create a new, unprivileged user on the system through the web interface. The second allowed any unprivileged user to escalate their privileges to root and write files to the filesystem. Combined, these two vulnerabilities enabled an attacker to take complete control of a vulnerable system.

It was estimated that prior to disclosure, over 10,000 systems were already infected with malicious code. Within a few days of disclosure, it was estimated that over 50,000 systems had been compromised and backdoored.

There were two interesting things about this vulnerability. First, the initial announcement from Cisco included a list of indicators of compromise (IOCs) – telltale signs that users could look for to see if their device had been breached. The inclusion of IOCs in an initial announcement is always a bad sign; it means that there is already a significant population of compromised devices.

Second, the initial vulnerability was through the administration web interface of the device. Security best practices require that such interfaces be disabled or reachable only from trusted sources. However this proved not to be the case for tens of thousands of systems that had their administration interface open to the public Internet.

Given the numerous Cisco devices in many organizations, security teams needed to be able to quickly locate the systems running IOS-XE specifically and then isolate those with their web interfaces exposed, enabling them to swiftly remediate these devices under the pressure of active exploitation. In response, runZero released **a query** that located IOS-XE devices with exposed administrative interfaces in a matter of hours.

## CASE STUDY: D-LINK NAS



When a zero day is identified, the clock starts ticking until the vendor releases a patch. But what if the patch never arrives?

In April 2024, **D-Link announced** a vulnerability in its DNS family of Network Attached Storage (NAS) devices, which enable users to store and share files over a network. These devices are EOL; the most recent model in this family ceased to be under support in 2020. Consequently, D-Link was unable to provide security updates and the official recommendation was to retire and replace these devices.

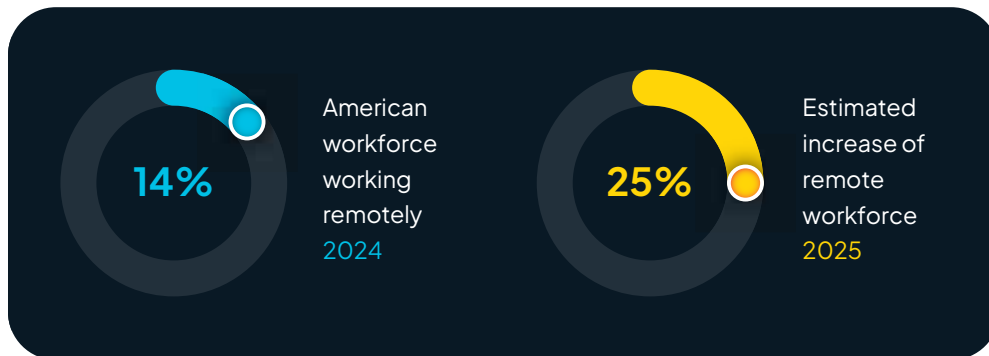
NAS devices have a long lifespan in homes and businesses, and are often used by small businesses without dedicated information security teams to enforce best practices. runZero research quickly located tens of thousands of these vulnerable devices exposed on the public Internet.

Because these sorts of devices tend to be deployed in small offices without dedicated staff, it's essential for organizations to detect when they are present on their networks. Failure to do so leaves the organization vulnerable to attack, making it important for asset inventory solutions like runZero to discover and highlight these vulnerable devices.

# Secure. Gateway. Pick One.

The COVID-19 pandemic accelerated the growth of remote work.

According to USA Today, 14% of the American workforce works remotely on a full-time basis and some estimate that number will increase to nearly 25% by the end of 2025.

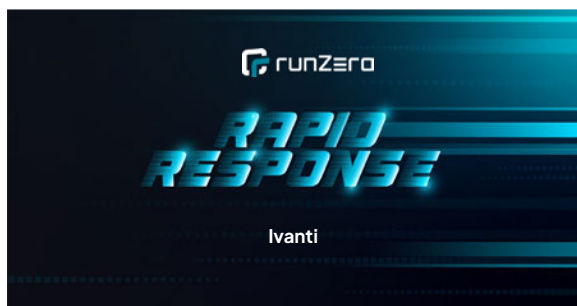


Organizations have leaned on secure gateway devices to bridge the moat between the secure corporate network and their remote employees, allowing remote workers to access corporate tools while working offsite.

Given their nature and purpose, secure gateway devices are extremely attractive targets to attackers. They must be exposed to the public Internet in order to function; they are often used to transmit secure information; and they are designed to allow (controlled) access to the secure inner network.

As such, when multiple critical vulnerabilities were discovered in these devices over the past several months, attackers wasted no time exploiting them. The blast radius from these attacks was widespread, with subsequent effects ranging from stolen personally identifiable information (PII) to persistence tools left behind by attackers to prolong exploitation efforts for long-term gain.

## CASE STUDY: IVANTI



Ivanti Connect Secure is a popular SSL-VPN solution, providing remote access from any web browser to internal organizational resources. Three times in the first half of 2024, in January, February, and April, Ivanti disclosed critical vulnerabilities in their Connect Secure systems.

The initial disclosure on January 10th was communicated with an ominous note that stated there was active exploitation of the vulnerability in the wild. As these systems have a heavy presence in government organizations and large corporations, exploitation poses a potential threat to national security.

By January 31st, the Cybersecurity and Infrastructure Security Agency (CISA), the agency responsible for the cybersecurity of the United States federal government's networks, issued an emergency directive requiring all federal agencies remove the affected Ivanti products from their networks by February 2nd. CISA indicated that two of their own systems were compromised as a result of these issues.

It is suspected that these vulnerabilities were exploited by adversarial nation-state actors affiliated with military, intelligence, or cybersecurity agencies.

## ENEMY AT THE GATES

The Ivanti attacks are not unique. Major vulnerabilities have been recently discovered – and exploited – in a large number of gateway and remote collaboration systems. In the first four months of 2024 alone, vulnerabilities and compromises were disclosed in gateway and remote collaboration systems from Fortra, AnyDesk, ConnectWise, Progress Software, and TeamCity.

Attackers are increasingly focusing their efforts on these gateway and remote collaboration systems, a departure from their historical focus on servers and client systems. If a gateway is present, it's a likely target.

**“Threat actors have recently developed workarounds to earlier mitigations and detection methods.”**





# Supply Chains Under Attack: Witness XZ Utils

On March 29th, the world woke up to what will likely (hopefully?) be the biggest security exposure of 2024. Microsoft engineer Andres Freund announced that a backdoor had been discovered in the xz-utils project.

xz-utils and its associated xz file format are extensively used in open source and proprietary software, and the tools and associated libraries are widely included in existing operating systems and applications. Notably in this context, the library implementing the compression algorithm, liblzma, is linked into various system services by several popular Linux distributions. The backdoor leaned on this linkage to allow authentication bypass in OpenSSH servers on these systems.

Had the backdoor not been discovered when it was, the downstream impact would have been catastrophic: popular Linux distributions, deployed in all sorts of environments, would have been open to attack and compromise.

Both fascinating and terrifying, the story behind how the xz-utils backdoor was planted is one for ages, involving sophisticated social engineering strategies that played out over several years to gain trust and then take advantage of that trust at opportunistic moments. The attack serves as a stark reminder that cyber criminals and nation states are increasingly innovative, advanced, and persistent in their techniques, and that they are willing to play the long game to reap significant rewards. This is particularly true when it comes to targeting supply chain attacks.

The long-term impact of a compromised supply chain is hard to quantify given the immense scope, making attacks of this nature one of the biggest threats we face today. We should all count ourselves lucky that the xz-utils backdoor was caught in the nick of time.

# runZero's Response to XZ Utils

Rated as critical with a CVSS score of 10.0, the entire information security world was united and immediately mobilized when the xz compromise was announced, recognizing the severity of a backdoor that could allow a threat actor to run arbitrary commands without authentication through vulnerable SSH daemons.

runZero's Rapid Response process was similarly invoked given the gravity of the situation, with the goal of helping security teams quickly identify potentially affected systems. To start, we knew that almost every Linux server would be running SSH of some sort, so simply saying "look at all these Linux systems running SSH" was not particularly helpful. Additionally, distributions like Fedora and Kali are often used in one-off installations, outside of the scope of regular IT controls – the so-called "shadow IT" of an organization.

We asked ourselves the \$65,535 question: how do we detect systems that could contain this backdoor, even the systems IT doesn't know about? To answer it, we found and installed the vulnerable versions of the compromised Linux distributions so that we could experiment with them. We then documented the versions of OpenSSH that corresponded to the versions in the vulnerable systems, as well as how OpenSSH reported itself in each of these environments (the "version exchange" that is sent by OpenSSH upon receiving a new connection).

To further narrow results, we leveraged runZero's novel operating system identification. Every operating system has slight quirks in how it talks on a network, using slightly different sets of values or behaviors that are within the bounds of the standard.



## // Research Note

Our research team maintains a compendium of "network protocol quirks" that can identify an operating system, often down to individual kernel versions and operating system releases, based solely on how these systems communicate on a network.

In this case, we built a comprehensive data set mapping eight different low-level network protocol stack fingerprints and OpenSSH banner values to Linux kernel versions and Linux distribution identifiers, which resulted in this simple query:

```
_asset.protocol:ssh
AND
protocol:ssh
AND
tcp.winScale:=7
AND (
    tcp.win:=31856
    OR
    tcp.mssMultiplier:=22
    OR
    tcp.mssMultiplier:=23
) AND (
    banner:="SSH-2.0-OpenSSH_9.6"
    OR
    banner:="SSH-2.0-OpenSSH_9.6p1%Debian%"
    OR
    banner:="SSH-2.0-OpenSSH_9.7p1%Debian%" )
```

The query can be described colloquially as “find OpenSSH versions that are new enough to be vulnerable, running on top of versions of the Linux kernel recent enough to have been part of the vulnerable releases.” It does this by looking for telltale low-level behaviors in the relevant Linux kernel versions’ network stacks, along with SSH pre-authentication banners, to find systems that fit the profile of potentially compromised systems.

This query will detect potentially vulnerable systems even if SSH is the only service they’re running, without the need to have endpoint management present or have the devices participate in a software inventory.

## // Research Note

This means potentially vulnerable devices could be located even if we didn’t know they existed in the first place.

## THE FALLOUT

In the end, how many vulnerable systems are there out there? We'll likely never know for certain, but the backdoor did make it into the real world. Systems were, and almost certainly still are, affected.

From the runZero perspective, we were able to look back at the asset data from just before the backdoor was discovered, and we found approximately 30 systems in disparate environments that were potentially affected and exposed to the Internet. Importantly, we were able to alert the security teams managing these systems, narrowing firefighting efforts down from hundreds of thousands of systems to 30 systems that needed immediate attention.

**Figure 19: A sample of matches illustrating the detected operating system and SSH version advertisement.**

Debian Linux	SSH-2.0-OpenSSH_9.6p1 Debian-5
Debian Linux	SSH-2.0-OpenSSH_9.7p1 Debian-2
Debian Linux 11.0	SSH-2.0-OpenSSH_9.6
Debian Linux 12.0	SSH-2.0-OpenSSH_9.6p1 Debian-3
Debian Linux 12.0	SSH-2.0-OpenSSH_9.6p1 Debian-4



# Fingerprints & Snapshots

Years ago, a “snapshot” was a photograph taken on the spur of the moment, without preparation. Nowadays, it means a moment frozen in time. While poetic, it’s also accurate: we can look at the state of security only as it was at a given moment, but never as it is now.

In this chapter, we dig deep into modern operating system fingerprinting through the lens of TCP/IP and four protocols critical to network security and system management. We present runZero’s observations on the state of Secure Shell (SSH) deployments, Transport Layer Security (TLS) stack demographics, the awkward state of the Remote Desktop Protocol (RDP), and review the long tail of the Server Message Block (SMB) protocol.

This snapshot provides some much-needed ground truth, especially in light of CISA’s **Binding Operational Directive 23-02**, which focuses on mitigating exposure from Internet-facing remote management interfaces.

# Fingerprinting Concepts

For the purposes of this section, “fingerprinting” is defined as the process of trying to identify, with as much precision as possible, some aspect of an asset. Fingerprinting techniques generally fall into one of three categories:



## Self identification:

The asset, via protocols, announces what it is.



## Attribute based:

Identification is accomplished via a set of observed values that are known to be unique to or indicative of a specific kind of asset.



## Behavior based:

Identification is accomplished by observing how the asset responds to certain stimuli.

There can be significant variation in the precision that can be achieved when fingerprinting. In one situation we may be able to identify the operating system and exact build number. In another case, it may only be possible to vaguely bucket the asset into an OS family such as “Windows” or “Linux.” Both outcomes can be possible against the same asset depending on which protocols and services we can observe.

# Operating System Fingerprinting

Identifying the operating system (OS) of a network-connected system, without credentials, and with minimal services, has always been a game of precision. Some of the trickiest examples are the forks of the Red Hat Enterprise Linux (RHEL) distribution. Often, the only real difference between these distributions is the replacement of Red Hat trademarks and branding with that of the particular Linux project. In many cases, these distributions are byte-for-byte identical, at the package level, and at the network level. These present a challenge to remote fingerprinting as a result.

To overcome these challenges, we collect and analyze enormous amounts of data. Our first pass at trying to differentiate the RHEL derivatives used a combination of two attributes, such as SSH version negotiation strings and the TCP Receive Window size. Over time, we realized this wasn't going to be sufficient and that we needed more and better data.

Analyzing data at scale is useful, but in situations like this it is vital to know exactly what combination of distribution and version leads to what results. For this effort we built hundreds of virtual machines running as many versions of the different distributions as we could. In some cases, these releases were over two decades old!

## // Research Note

CentOS and certain other Linux distributions such as Oracle Linux were originally forks or “bug and binary compatible” redistributions of Red Hat Enterprise Linux. The relationship changed in 2021 when Red Hat, which acquired CentOS in 2014, discontinued CentOS Linux and created CentOS Stream. With this change CentOS would no longer be downstream of RHEL but would instead be the upstream source from which RHEL is created. The logical flow now has Fedora as the root with both CentOS and RHEL downstream. In response to CentOS Linux being discontinued two new distributions were created: AlmaLinux OS and Rocky Linux.

## VERIFY TARGET, ONE SYN ONLY

From each of these virtual machines we collected as much information as we could about how the TCP stack communicated. While it is true that fingerprinting an operating system via TCP stack quirks has been a thing for years, our challenge was to improve our detection while sending the absolute minimum amount of traffic and, importantly, to look for evidence that would persist through common configuration changes by the system administrators.

To explain our findings, we first need to define some terms:

→ **TCP Receive Window:**

Maximum amount of data that a particular endpoint can receive and buffer. The sending host has to stop after sending the maximum amount of data and wait for ACK and window updates.

→ **MTU:**

Maximum transmission unit, which is the largest packet that the network interface can accept.

→ **MSS:**

Maximum segment size, which is the maximum amount of TCP data that can fit into a single packet, calculated as the MTU minus the protocol headers.

→ **TCP Window Scale:**

An optional factor by which the TCP Receive Window is scaled; this allows receive windows to exceed the maximum of 65535 bytes that can be specified in the TCP Receive Window field.

Of the TCP attributes that we observed, the one that provided the murkiest fingerprinting results was the TCP Window Scale. The values for it, when present, range from 0 to 14. With this information, we can usually determine if the target is running a general family of operating systems.

**Figure 20: TCP Window Scale by Operating System.**

TCP Window Scale	Operating System
2	Linux 2.x prior to 2.6.18
5/6	BSD-based systems, including Apple MacOS, iOS, iPadOS, and tvOS
7	Linux 2.6.18 and higher
8	Microsoft Windows and miscellaneous others such as Roku OS
9	VMware ESXi, certain embedded Linux devices



Combining the TCP Receive Window and MSS offered the next significant improvement. In our past work, leveraging the Receive Window size sometimes yielded values that seemed to change unexpectedly. The reason why became clear when we looked at the data from the lab. The key points were:

- Changes to the link-layer MTU impacts the value of MSS, since MSS is calculated as the MTU minus the size of certain TCP/IP headers.
- MSS is different between IPv6 and IPv4 due to the IPv6 IP headers being 20 bytes larger.
- For Linux-based systems, Receive Windows less than the maximum value were almost always an even multiple of MSS. Due to the MSS difference mentioned above this means that the Receive Windows would vary as well.
- Critically, the MSS multiplier for Linux-based OSes correlated with the Linux kernel version.

With the information above in hand, we can organize Linux systems into specific kernel version buckets based on the observed multiplier. That is quite a bit of information from the response to a single SYN packet!

**Figure 21: A table indicating the relationship between IPv4/IPv6 MSS Multiplier and Linux Kernel version.**

IPv4 MSS Multiplier	IPv6 MSS Multiplier	Linux Kernel
4	4	2.x to 2.6.32-131
10	10	2.6.32-220 to 3.10.229
20	20	3.10.0-327 to 4.18.0
45	45	5.0 to 6.5.x
22	23	6.6 to current

The kernel version also offers a hint as to the relative age of the system. A MSS multiplier of 4 indicates that the machine is likely running an ancient version of Linux, far beyond EOL, and certainly not something that should still be in production.

## A LITTLE FROM COLUMN A, A LITTLE FROM COLUMN B

TCP-based fingerprinting by itself doesn't improve fingerprinting of RHEL derivatives as much as we'd like. Since most of the systems in our analysis had SSH running, we looked for patterns in RHEL-derivative type and version in the light of SSH version negotiation advertisements (for example, SSH-2.0-OpenSSH\_8.7) combined with the Linux kernel version. This strategy quickly yielded results. We found that we could generally identify the distribution's major version, and in some cases, minor version range as well.

The screenshots below demonstrates how specific patterns pop out under bulk analysis.

**Figure 22: A table illustrating the relationship between different Enterprise Linux distribution versions and various network attributes.**

Platform	Version	SSH banner	v4 tcp.win	v4 MSS	MSS Multiplier	v4 Window Scale
CentOS Linux	7.1	SSH-2.0-OpenSSH_6.6.1	14480	1460	10	7
CentOS Linux	7.2	SSH-2.0-OpenSSH_6.6.1	28960	1460	20	7
CentOS Linux	7.3		28960	1460	20	7
CentOS Linux	7.4		28960	1460	20	7
CentOS Linux	7.5	SSH-2.0-OpenSSH_7.4	28960	1460	20	7
Oracle Linux Server	7.7		28960	1460	20	7
CentOS Linux	7.9		28960	1460	20	7
Oracle Linux Server	7.9		28960	1460	20	7
Scientific Linux	7.9		28960	1460	20	7
CentOS Linux	8.0	SSH-2.0-OpenSSH_7.8	28960	1460	20	7
Oracle Linux Server	8.0		28960	1460	20	7
CentOS Linux	8.1	SSH-2.0-OpenSSH_8.0	28960	1460	20	7
AlmaLinux	8.9		28960	1460	20	7
Red Hat Enterprise Linux	8.9		28960	1460	20	7
Rocky Linux	8.9		28960	1460	20	7
Oracle Linux Server	8.9		28960	1460	20	7
Oracle Linux Server	8.7		SSH-2.0-OpenSSH_8.0	65160	1460	45
Oracle Linux Server	8.9	65160		1460	45	7
Oracle Linux Server	9.1	SSH-2.0-OpenSSH_8.7	65160	1460	45	7
Oracle Linux Server	9.2		65160	1460	45	7
Rocky Linux	9.2		65160	1460	45	7
AlmaLinux	9.3		65160	1460	45	7
Oracle Linux Server	9.3		65160	1460	45	7
Red Hat Enterprise Linux	9.3		65160	1460	45	7
Rocky Linux	9.3		65160	1460	45	7

As we can see in this screenshot, by combining SSH version advertisement and various measured TCP quirks, it is possible to narrow the Linux distribution involved, often down to individual point releases. Even when it is not possible to precisely determine the version, it is almost always possible to determine if the distribution in question is derived from RHEL.

Figure 23: runZero detecting operating systems derived from Red Hat Enterprise Linux.

Up	OS	Type
●	RHEL Derivative Linux 3	Server
●	RHEL Derivative Linux 7	Server
●	RHEL Derivative Linux 7.0	Server
●	RHEL Derivative Linux 7.1	Server
●	RHEL Derivative Linux 8	Server
●	RHEL Derivative Linux 8.0	Server
●	RHEL Derivative Linux 9	Server
●	RHEL Derivative Linux 9	Server

## Secure Shell

The Secure Shell (SSH) protocol is an encrypted network protocol used to access an interactive shell and perform file transfers between systems over untrusted networks. SSH is the de facto management protocol for non-Windows machines (and even some Windows systems), replacing the Telnet protocol from days past.

The most recent version of the protocol, **SSH-2**, was standardized in 2006 and provides a high level of security when configured correctly. runZero analyzed aspects of SSH in the ecosystem to explore how SSH is being deployed in the real world.

## MY VOICE IS MY PASSPORT, VERIFY ME.

The Secure Shell protocol consists of three phases. First, a secure transport is negotiated, similar to TLS. After the transport key negotiation is complete, the client attempts to authenticate, specifying one of a handful of known methods, and the server replies indicating whether the authentication succeeded and what remaining authentication methods are available if not. Finally, after successful authentication, a session is opened. This session enables access to channels, which in turn provide interactive shells, port forwarding, agent forwarding, and file transfer capabilities, among other options.

The three most common authentication mechanisms are:



### Password:

Traditional username-plus-password authentication.



### Keyboard-interactive:

Accommodates multiple back-and-forth steps to support additional challenges, like one-time passwords or multi-factor authentication tokens; however, the presence of keyboard-interactive doesn't guarantee stronger authentication as many systems are configured to treat it effectively the same as password authentication.



### Publickey:

Uses public-key cryptography, where the client provides proof of the private key by signing a challenge from the server.

Of the three mechanisms, publickey is by far the most secure and considered best practice. This type of authentication also supports key certificates, which provide even stronger security for key issuance and revocation.

In publickey authentication, a user's public key is stored in their profile on the destination system and only someone with the corresponding private key can authenticate. This prevents compromise through password-guessing attacks.

**Figure 24: A partial screenshot of runZero showing the results of an SSH scan.**

192.168.86.32 - 22/tcp	
banner	SSH-2.0-OpenSSH_8.7
fp.certainty	0.85
product	OpenBSD.OpenSSH:8.7
protocol	ssh
source	connect
ssh.authMethods	password · publickey
ssh.authPassword	true
ssh.authPublicKey	true

In our survey of SSH endpoints, **54% support both password and publickey authentication**. This is the default for many modern SSH services, and allows the optional use of a strong public key while allowing for the ease of password setup. In general, best practices recommend that the password mechanism be used only to set up public key authentication, after which it should be disabled. **Leaving password authentication enabled exposes systems to password enumeration attacks and the potential for user-set weak passwords.**

Password authentication is more common on storage and networking devices, where accounts are less likely to be associated with individual persons. Often, though, these systems still support publickey authentication, which should always be preferred over password authentication where possible.

Looking at the big picture, 95% of SSH endpoints offer publickey authentication. Whether these systems are configured to use it is another question entirely. What is perhaps somewhat disheartening is that, while 95% of endpoints support the most secure mechanism, approximately 92% still support some form of password authentication as well.

**Figure 25: Distribution of SSH authentication method combinations.**

password publickey	53.91%
keyboard-interactive password publickey	23.67%
keyboard-interactive publickey	9.81%
publickey	8.02%
password	3.32%
keyboard-interactive password	1.17%
keyboard-interactive	0.09%

## SSHIP OF THESEUS

SSH servers identify themselves by way of an SSH host key pair. Just as key pairs allow users to prove their identity, so too do host keys allow servers to prove their identity to users.

This functionality is critically important. Without it, users could be tricked into thinking that they had logged into a totally different, possibly spoofed or malicious, system, with obvious security ramifications. However, unlike TLS, most SSH servers are not configured to use any form of Public Key Infrastructure (PKI) or other chain-of-trust to establish proof of server identity. Such functionality is available, but is not in widespread use.

Instead, most SSH clients will use a technique called Trust on First Use (TOFU). In this scheme, the client will trust a host key the first time it receives it for a given host. Going forward, if the host key changes, the SSH client can alert the user to the problem. While this doesn't allow the user to confirm the host's identity, it at least allows them to confirm that the host's identity hasn't changed.

```
ubuntu@u2404-infra-02:~$ ssh 192.168.50.127
The authenticity of host '192.168.50.127
(192.168.50.127)' can't be established.
ED25519 key fingerprint is SHA256:wdNLQA-
2vyp6Qv+8T7Ac2rF6vRJz34P5RCQo9VJAa+Ms.
This key is not known by any other names.
Are you sure you want to continue connecting
(yes/no/[fingerprint])? █
```

While host keys are ostensibly used to uniquely identify a host, oftentimes multiple hosts have the same host key. This is sometimes intentional, such as when automatically provisioning many ephemeral systems. Unfortunately, it can also happen accidentally and this can have very undesirable consequences.
















Or at least prove their possession of the correct private key.

## // Research Note

We performed a limited audit to see how frequently host keys were being reused across our data set. We identified more than 350 instances where the same host key was shared across unrelated environments. Further exploration across the wider Internet revealed thousands of additional shared host keys.

The good news is that most of these situations can be avoided by using tools that are already available. SSH certificate-based authentication, which allows a trusted certificate authority to sign host keys, can provide clients with assurance that a never-before-seen host key can be trusted.

**Figure 26: Reuse of individual keys across our data set by device type.**

<b>Key 1</b>	Seen on 7,400+ servers		<b>Key 11</b>	Seen on 1,200+ servers	
<b>Key 2</b>	Seen on 4,300+ storage appliances		<b>Key 12</b>	Seen on 1,100+ servers	
<b>Key 3</b>	Seen on 3,800+ servers		<b>Key 13</b>	Seen on 1,000+ servers	
<b>Key 4</b>	Seen on 3,200+ servers		<b>Key 14</b>	Seen on 1,000+ servers	
<b>Key 5</b>	Seen on 1,900+ servers		<b>Key 15</b>	Seen on 950+ servers	
<b>Key 6</b>	Seen on 1,700+ servers		<b>Key 16</b>	Seen on 900+ servers	
<b>Key 7</b>	Seen on 1,500+ servers		<b>Key 17</b>	Seen on 850+ servers	
<b>Key 8</b>	Seen on 1,400+ servers		<b>Key 18</b>	Seen on 850+ servers	
<b>Key 9</b>	Seen on 1,300+ servers		<b>Key 19</b>	Seen on 750+ servers	
<b>Key 10</b>	Seen on 1,200+ servers		<b>Key 20</b>	Seen on 700+ servers	

# Transport Layer Security

TLS, or **Transport Layer Security**, is the de facto standard for encrypted communications over the Internet. It is responsible for securing and validating communication between two different systems across untrusted networks.

The runZero scanner fingerprints TLS implementations (the “stack”) automatically as part of the discovery process. This feature was added in October 2022 to help customers identify OpenSSL 3.0.x endpoints in response to an announcement that a critical vulnerability was present in these versions, but not in the older 1.1.1 release. runZero was able to help customers identify their OpenSSL 3.0.x services before the details of the issue were made public and has continued to improve this functionality ever since.

Although there are a handful of existing TLS fingerprinting implementations, runZero decided to build something new for three reasons:

- Existing techniques were not resilient to communication tampering by devices along the network path.
- Existing techniques were overly reliant on attributes that can and do vary naturally due to configuration choices by administrators.
- Existing techniques used “lossy” hashing that limited their utility to lookup tables.

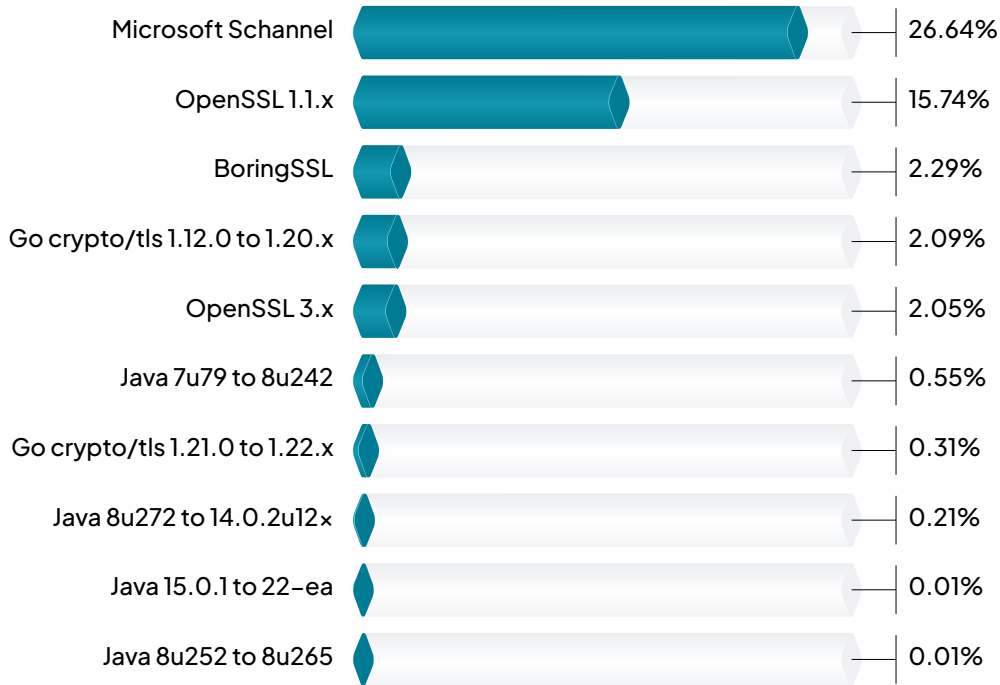
runZero’s initial research in this area focused on trying to find protocol “quirks” that could identify individual TLS stacks. For example, what does the stack do when sent an empty value for the *Server Name Indication* extension? Does it respond with a *Server Hello* or an *Alert Message*? There are at least five possible responses in this situation, and each one helps narrow down which TLS stack it might be.

As part of this effort, runZero audited and collated the responses from as many libraries (and versions of those libraries!) as possible. In many cases, lab test implementations were built to ensure the work could be replicated and validated. Identifying characteristics were then pared down to the minimal possible sets to ensure the scanning process sent as little traffic as possible to customer devices.





**Figure 27: Top 10 fingerprinted TLS stacks.**



As a result of this work runZero can identify many common (and not so common!) TLS stacks, often down to specific library versions. This information can prove invaluable in identifying not just the TLS stack, but also with detecting vulnerabilities and EOL systems.

**Figure 28: Screenshot showing TLS fingerprint attributes in field `tls.rzfp0` and the resulting TLS stack assertion in `tls.stack`.**

Attribute	Value
tls.rzfp0	v0 t10:alert#02#46:,t12:hello#0303#c02f#:sdone#00#,t13:hello#0304#1301#002b=n0002/0033=n0024:hello#0000#0000#,f0:hello#0304#1301#002b=n0002/0033=n0024:hello#0000#0000#,f5:hello#0304#1301#002b=n0002/0033=n0024:hello#0000#0000#,sne:alert#02#0a:,t12hc:hello#0303#c02f#:sdone#00#,ale:alert#02#0a:,alu:alert#02#78:,fd2:alert#02#28:,rts:random 0304 fd2  fd2
tls.serial	ccae983745b38873542a47ac9c9ade1b
tls.stack	go-crypto-tls=1.21.0^1.22.1
tls.supportedVersionNames	TLSv1.2 TLSv1.3

Note that the fingerprinting was able to identify the version of Go's TLS implementation to high precision.

## CASE STUDY: OPENSSL 1.1.1 IS END-OF-LIFE

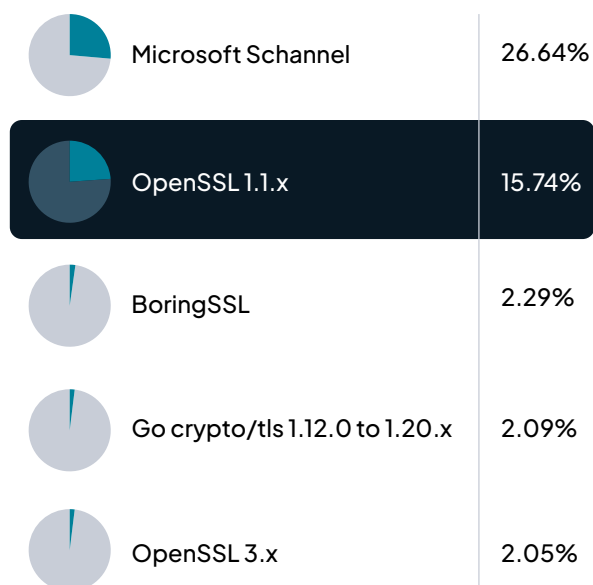
The TLS stack fingerprint can do double duty, serving as a useful indicator for the age of the associated system. For example, OpenSSL 1.1.1 reached end-of-life in September of 2023. That means that, in general, no further fixes, security or otherwise, will be made for that version of the library. Since that date, there have been two publicly-acknowledged vulnerabilities in OpenSSL 1.1.1.

runZero's analysis shows that **OpenSSL 1.1.1 is still present on roughly 16% of TLS services**. It shows up everywhere: routers, switches, printers, phones, cameras, and power devices, amongst many others. These systems either weren't or can't be upgraded to newer versions and that means roughly one out of every six TLS services is using a library that will no longer receive security updates.

The ability to fingerprint TLS stacks directly is critical because the OpenSSL version used by a service is not typically visible to the average user or administrator. Many systems include multiple TLS implementations, so even the presence of old OpenSSL shared libraries on a system wouldn't necessarily say which services would be affected. To complicate things further, some services have statically linked the OpenSSL library, and the only way to identify their use is to examine the service binaries or communicate with them directly.

runZero solves this challenge by measuring how the service actually communicates, providing an efficient way to find systems that could be stuck with a version of OpenSSL that will never see a new version.

Figure 29: Top 5 fingerprinted TLS stacks.



Roughly one out of every six TLS services is using a library that will no longer receive security updates.

# The Remote Desktop Protocol

In the early 2000s, Microsoft's Remote Desktop Services (RDS, then called Terminal Services), presented a few security challenges: clients couldn't validate the server's identity, there was no brute force prevention, and unauthenticated users could connect and observe the login screen. The login screen often displayed a default username and the domain that the server was part of. This information could then be used in brute force attacks.

**Figure 30: Microsoft's Remote Desktop Services login screen.**



Additionally, upon initial connection, the server would provision an entire desktop environment before beginning the login process. This meant that attackers could easily create a resource-exhaustion situation by simply opening a large number of sessions.

With the release of Windows 2003 Service Pack 1, Microsoft introduced the ability to use TLS, which addressed the issue of machine-in-the-middle (MitM) attacks by allowing clients to cryptographically verify they were connecting to the expected server.

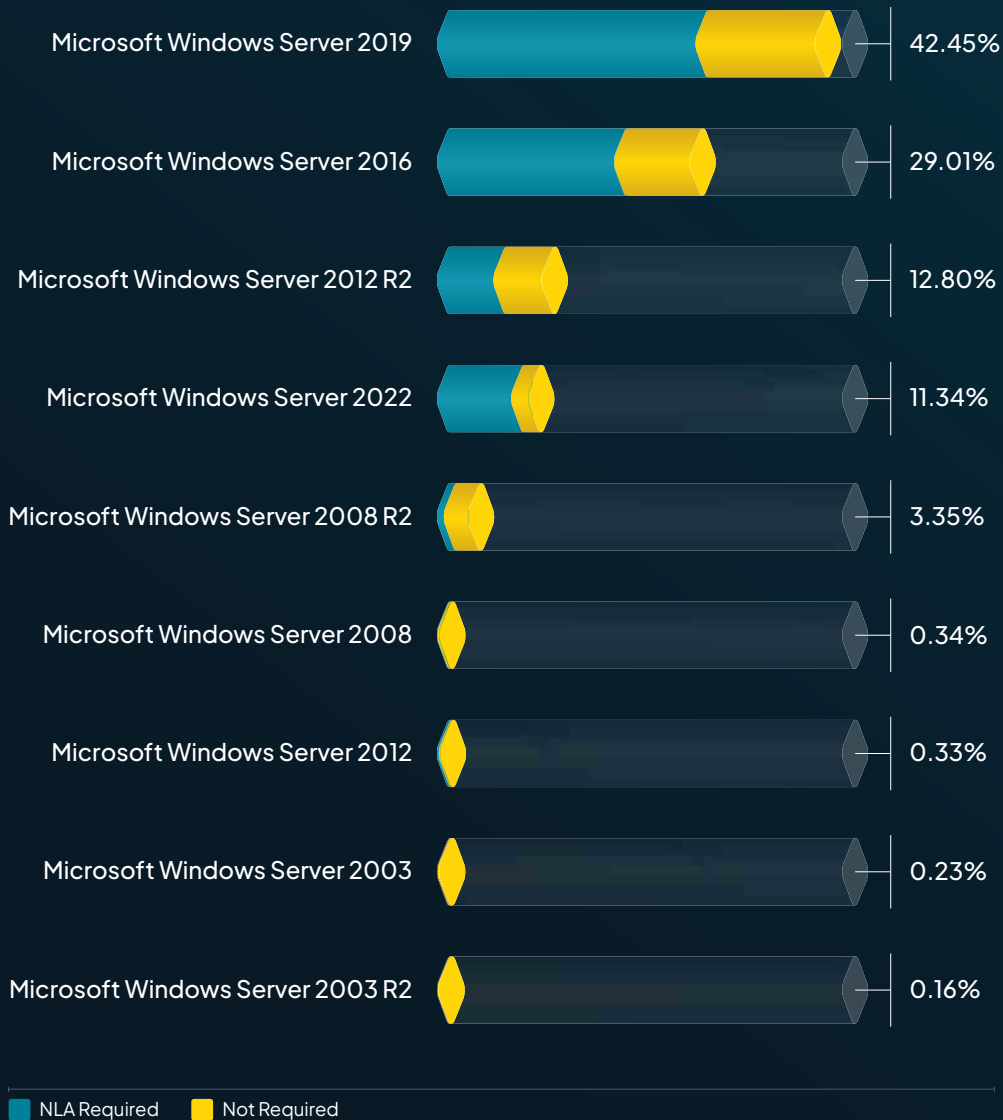
In Windows Server 2008, Microsoft introduced Network Level Authentication (NLA), which required users to authenticate themselves before a session would be established. NLA forced authentication to occur after the TLS handshake, but before the console was provisioned, which mitigated the resource-exhaustion concerns, reduced information leakage, and significantly impaired brute-force attacks.

When configuring RDS in Windows Server 2008, administrators had the option to require NLA for all connections or to allow the client to decide. Starting with Windows 2012, however, NLA was required by default to improve security.

We explored how frequently organizations choose non-default options for NLA enforcement. As the results illustrate, the majority of Remote Desktop services on Windows Server versions where NLA is required by default do, in fact, require NLA. This is good news, and an indicator that secure defaults can have a positive impact.

An argument could be made that the NLA requirement being disabled by default on Windows Server 2008 / 2008 R2 shows up in the results as well, but this state may be influenced by those servers being more likely to have legacy or third-party clients that don't support NLA.

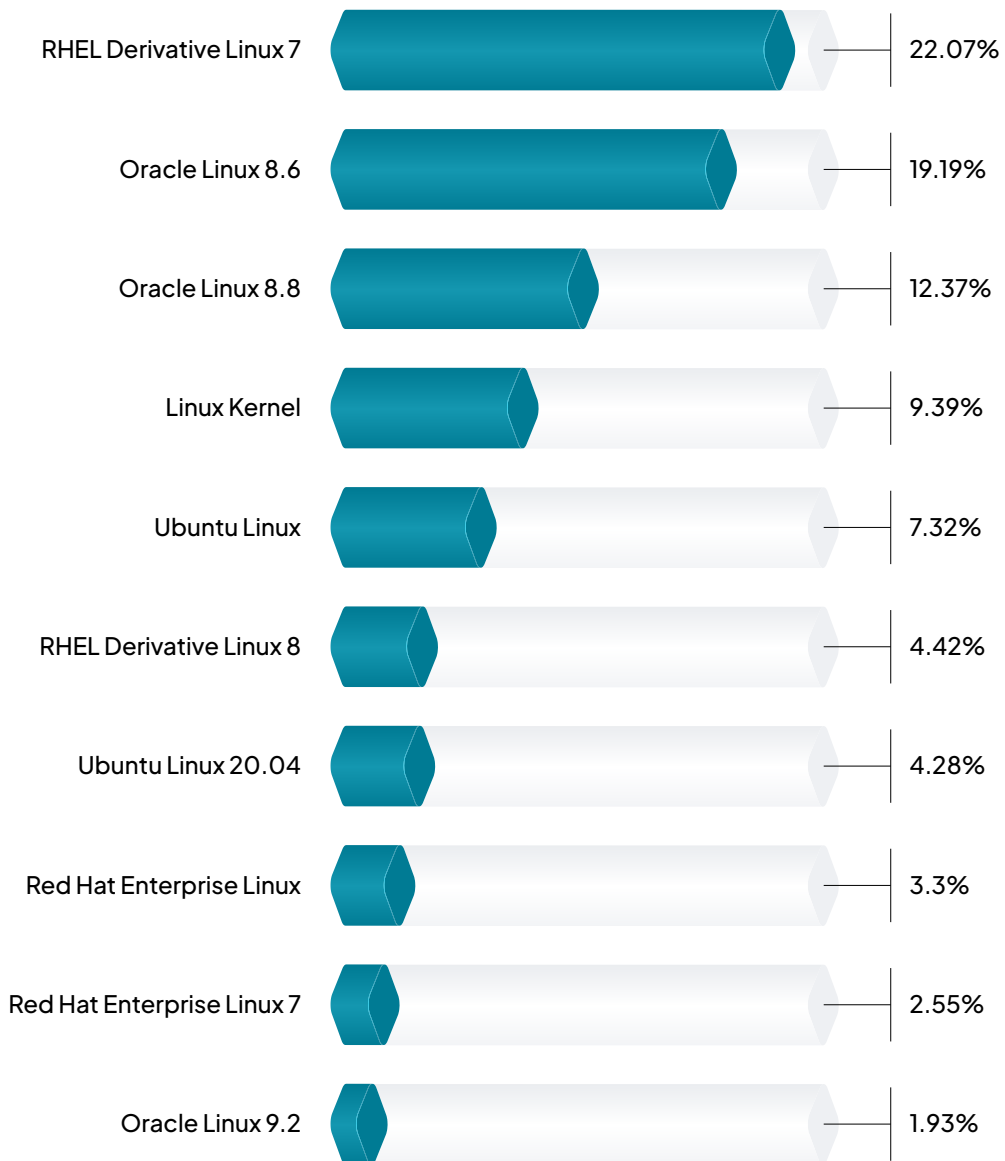
**Figure 31: RDP NLA enforcement – OS distribution.**



We also reviewed the OS distribution of services that did not permit using NLA at all. This list is dominated by Red Hat Enterprise Linux and its various derivatives running the xrdp RDP service.

The xrdp service does not support NLA, so these results are not surprising. However, we were encouraged to find so few results for Microsoft Windows machines without NLA support that the number is not statistically significant. This implies that secure defaults work.

**Figure 32: RDP without NLA support – OS distribution.**



# Server Message Block

The Server Message Block (SMB) protocol is used by Microsoft Windows for remote file access, printer sharing, and a laundry list of remote management features. SMB has a history stretching back to 1983, originally developed at IBM for file and printer sharing. Given its age, SMB has been both the target of, and vector for, countless attacks by malicious actors.

The protocol has evolved greatly over the years: from a protocol with limited security targeting small, low-speed local networks in the early days, to a secure and featureful protocol with relatively good performance over the wide area networks.

Security documentation commonly recommends disabling SMB version 1. SMBv1 has been superseded over the years by multiple versions of SMB v2 and SMB v3, both of which provide major security improvements.

SMBv1 does not support encryption of data nor protection from protocol security downgrade attacks or MitM tampering. While it does include message signing for some messages, these signatures use the legacy MD5 function, which was itself replaced by the much stronger HMAC SHA-256 in SMBv2 and AES-CMAC in SMBv3.

Newer SMB protocols have stronger security fundamentals and have also avoided severe vulnerabilities (such as [EternalBlue](#)) that only impacted SMBv1.

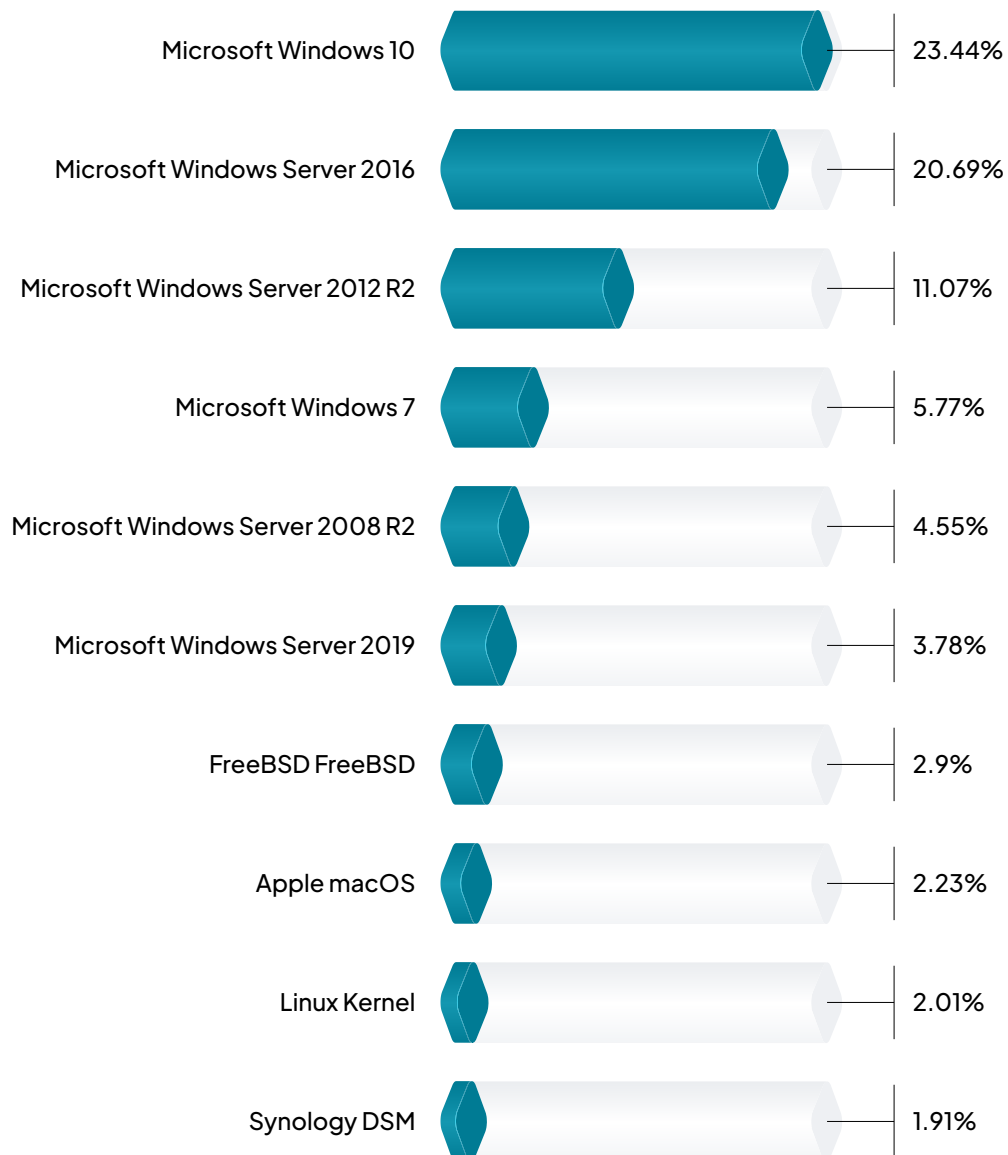
The last versions of Windows to actually require SMBv1 were Windows XP and Windows Server 2003, both of which were removed from support nearly a decade ago. SMBv1 was publicly deprecated in mid-2013 and Microsoft removed it entirely from clean installs of Windows 10 and Windows Server 2019 starting in late 2017. Microsoft even went so far as to automatically uninstall SMB v1 support from updated systems that had not used the older protocol during a 15-day sampling period.



## // Research Note

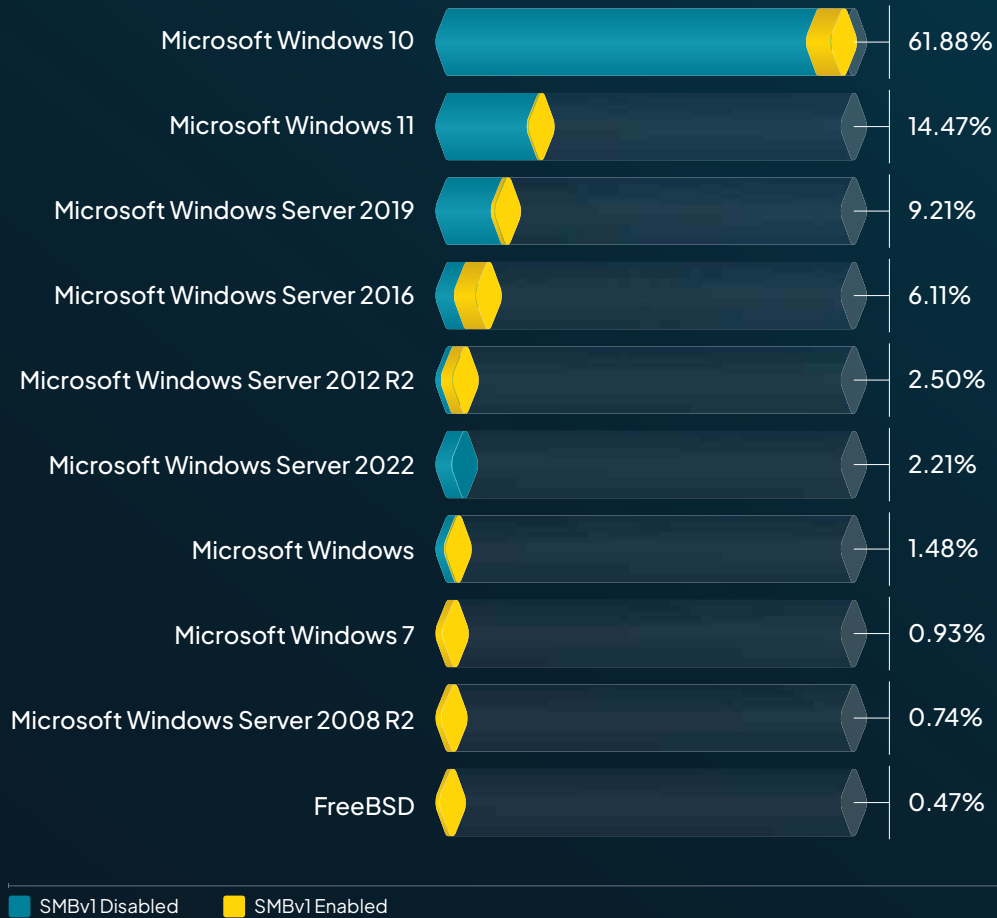
runZero analyzed the distribution of operating systems that still have SMBv1 enabled. While the number of services we observed was not exactly thrilling, it was good to see that, statistically, most instances were related to versions of Windows that may have been installed or upgraded while SMBv1 was still enabled by default, or in third-party systems that need to interact with the Windows ecosystem.

Figure 33: Percentages of OS (top 10) with SMBv1 enabled.



The SMB v1 statistics look a bit better once you break these down by operating system.

**Figure 34: SMBv1 enabled – distribution within OS.**



While it could be argued that the lack of SMBv1 in more recent systems is due to reduced need to interoperate with legacy systems, we'd still expect to see it more frequently if enabled by default. Microsoft's choice to disable SMBv1 by default seems to have had a significant positive impact on the security posture of environments with SMB present.





# AI & the Need for Specificity

## Exact answers are seemingly passé.

Everywhere we look, statistical models and neural networks have blossomed. Seemingly overnight, LLMs and other AI technologies have grown from fascinating curiosities to being embedded in everything, everywhere. Chatbots now handle customer service requests and teach foreign languages while large language models write dissertations for students and code for professionals.

Software companies are claiming – and seem to be realizing – gains in programmer productivity thanks to code generation by LLM-backed AIs. Language learning tools and automated translation have been revolutionized in just a few short years, and it is not hard to imagine that in the near-future, advanced artificial intelligence will be as commodified as the once-world-shaking smartphone.

AI tools provide answers that sound good and are easy for humans to consume, but struggle with a key challenge; knowing the truth. This flaw is a serious roadblock to using AI in security-critical workflows.

### // Our Perspective

Modern AI is undoubtedly a fascinating and powerful set of technologies, but these tools are ill-suited to CAASM, asset inventory, and vulnerability discovery efforts. runZero believes that current-generation AI is not just unhelpful for most security efforts, but can be actively harmful.

# Verification Is Everything

LLMs have proven excellent at prediction and generation, but struggle to provide useful outcomes when the workload requires high levels of precision.

In the case of content and code generation, LLMs do well because the user can quickly verify that the output matches the intent. Does the sentence make sense? Does the code compile? These are quick tests that the user can apply to determine whether the LLM provided an accurate response.

LLM-generated data presents two problems for CAASM and asset inventory:



There is no guarantee that the claims made by the tool are accurate, or even that the specific assets or vulnerabilities exist. Careful, prompt engineering might help, but it might not.



The inference mechanisms are black boxes. There is little way to know how the detected devices relate to the provided evidence or what was skimmed over or omitted by the inference process.

In short, without an efficient way to verify the output from an LLM, it is difficult to rely on these systems for discovery automation at scale.



## Slightly Wrong Is Rarely Right

LLMs struggle with another aspect of information security; the sheer scale of data. Even an AI tool that is 99% accurate at detecting vulnerabilities and classifying assets may result in worse outcomes than not using the tool at all. A one percent gap may seem small, but modern organizations manage asset and vulnerability records in the millions and even billions.

Meaningful exposures already exist in the margins of massive datasets. For every 1,000 workstations, there may only be one exposed system; however, that system might be the single entry point an attacker needs to succeed. For situations that require knowing exactly what and where things are, systems that provide exact answers are, well, exactly what is needed.

## Lies, Damn Lies, & Statistics

Statistical methods are beautiful applications of mathematics based on centuries of meticulous work, but the outcomes of these methods tend to be aggregate views and trends over time. Statistical models and AI tools built on these models, are great at providing high-level views, but unfortunately tend to bury the most critical exposures instead of flagging them for remediation efforts.

A great example of this is the average asset risk metric: does a single high-risk asset actually present the same risk as 10 low-risk assets? In almost all cases, the answer is no. There are times when we want to analyze generalities from the details because statistical methods are indispensable tools when it comes to reporting, overall distribution, and location of outliers. However, when we want to see exactly what assets exist, where they are, and what they do, statistical methods are less useful.

# Precision Matters

The goal of CAASM is to provide comprehensive and precise visibility into the entire organization, with a focus on minimizing exposure. The current-generation of AI tools struggle to help due to the outsized effort required to verify their results. Defenders already struggle with a deluge of noise from their tools and adding more wrong answers has a real human cost.

Statistical models, while helpful for measuring trends over time, also tend to obfuscate the most critical exposures in noise. CAASM requires precision at scale and failing to identify even one percent of an attack surface or an organizations' assets, is not an acceptable error rate. AI tools may be helpful for report generation and data summarization, but struggle to provide the level of accuracy required to deliver on the promise of CAASM.



**runZero delivers the fastest, most complete security visibility possible, providing the ultimate foundation for successfully managing risk and exposure.**

Rated number one on Gartner Peer Insights, our cyber asset attack surface management (CAASM) platform starts delivering insights in minutes, discovering both managed and unmanaged devices across the full spectrum of IT, OT, IoT, cloud, mobile, and remote assets.

Combining powerful, proprietary active scanning, passive discovery, and integrations enables runZero to deliver the most accurate, in-depth data and insights into everything on your network, without the need for credentials, agents, or hardware.

With a world-class NPS score of 82, runZero has been trusted by more than 30,000 users to improve security visibility since the company was founded by industry veteran HD Moore.

Connect with us:



Copyright © 2024 runZero, Inc. runZero is a registered trademark of runZero, Inc. runZero Explorer and 'Get to know your network' are trademarks of runZero, Inc. All other trademarks are properties of their respective owners.

Test drive the runZero Platform for 21 days, with an option to convert to our free Community Edition at the end of your trial.

**Try runZero for Free**

[runzero.com/try/](https://runzero.com/try/)