Zero-Trust

From Aspirational to Overdue

J.C. Checco, C|CISO, CISSP, CSSLP, CCSK, SDRM, MBA
Resident CISO, Financial Services – Proofpoint

Abstract

Security is a resiliency model not an efficiency model, and as such, many organizations have increased their Year-over-Year (YoY) spending on security technologies. But as the threat landscape has been evolving to more targeted people-centric TTPs; the incremental costs of bolt-on security solutions have less of an effective impact resulting in a YoY decreased Return on Investment (ROI) in security spending. If this trend continues, the ROI on security spend will eventually be negative.

The Zero Trust paradigm re-thinks an organization's security posture in terms of people-centric threats and data-centric protections, emphasizing the age-old security tenets of "least privilege", "segregation of duties" ,"continuous verification" and "security by design". By embedding security at the core infrastructure, Zero Trust introduces a holistic, long-term, resilient, dynamic and cost-effective approach to an organization's security posture.

Embarking on a new security paradigm such as Zero Trust can be daunting, unless you can learn to identify and prioritize where change will be most effective, most disruptive, and most challenging. Zero Trust is not a technology solution, it is a business mindset. And although Zero Trust may be a maturing security paradigm, the architecture and implementation can take advantage of existing security controls.

There is no "one size fits all" approach to a Zero Trust Initiative; hence we seek our peers – both partners and competitors – to discuss a thoughtful approach to executing (and operating) effectively in this new paradigm.

Contents

ABSTRACT **2**

EXECUTIVE SUMMARY **4**

EXPLAINING ZERO-TRUST TO YOUR GRANDMOTHER **5**

- Know Your Neighbor (Continuous Verification) 5
- Fire Prevention (Compartmentalization) 6
- Finding the Right Balance (Orchestration) 7

ZERO-TRUST REDUX **8**

- A History of Disruption 8
- Perimeter Security: One Size Fits All 9
- What was Old is New Again 9

WHY ZERO-TRUST **12**

- Aspirational 12
- Overdue 13

DEFINING ZERO-TRUST **16**

- Prerequisites 16
- Foundational Components 22
- Myths, Misconceptions & Assumptions 37
- The Forgotten Dependencies 39

ROADMAP TO ZERO-TRUST **52**

- Pre-Planning Steps 53
- Determining Good System Design: The Chainsaw Approach 56
- Maximizing Existing Security Investments 60
- Execution Plan 65

ZERO-TRUST TAKEAWAYS **71**

- Recommended Reading 72

ABOUT THE AUTHOR **73**

ABOUT PROOFPOINT **74**

ENDNOTES & REFERENCES **75**

Executive Summary

Over the past two decades, enterprises have slowly migrated from traditional offices to an open office paradigm, which provided more workforce density thus lowering the costs per square foot of commercial real estate. In a blink of an eye, though, these enterprises have had to move away from corporate office space to an all-remote workforce. Traditional perimeter protections have been rendered useless, and organizations are quickly attempting to build secure remote working environments. Simply expanding the VPN capacity has not proven successful. This whitepaper will discuss the accelerated efforts to rebuild the enterprise infrastructure into a zero-trust architecture – rebuilding the plane's engines while it is flying at 50,000 ft.

NIST's SP800-207 publication "Zero Trust Architecture (2nd Draft)" succinctly define: *Zero trust (ZT) is the term for an evolving set of cybersecurity paradigms that move network defenses from static, network-based perimeters to focus on* ***users***, ***assets***, *and* ***resources***. *A zero-trust architecture (ZTA) uses zero trust principles to plan enterprise infrastructure and workflows.*[1]

The key concept to remember is the mindset of "*zero trust principles*" when designing a security model for an enterprise. The NIST document's authors are correct in stating that: *ZT is not a single-network architecture but a set of guiding principles in network infrastructure and system design and operation that can be used to improve the security posture ….*[2]

There are several key ideologies that will be emphasized in this paper:

- promoting **Holistic Change**,
- defining **Asset** as the **Perimeter**,
- categorizing assets as **Actors, Conduits, Data** and **Workloads**
- building **Orchestration**,
- embracing the **Journey** and the **Destination,**
- exercising **Good Design Principles**, and
- attaining residual **Security Maturity.**

Our goal is to make a Zero-Trust implementation in a legacy enterprise more grounded, prevent feature creep as well as explain the infeasibilities of idealism.

Explaining Zero-Trust to Your Grandmother

For those well-versed in the nuances of what Zero-Trust entails, you can skip this section. For those looking to have an interesting party conversation, this can help the layperson understand what Zero-Trust is about. Instead of explaining the “zero” in ZT, just talk about three everyday concepts: knowing your neighbor, fire prevention and what makes your home safe.

Know Your Neighbor (Continuous Verification)

In any neighborhood, you see and chat with the same people walking by day after day. You know them by face, and you know where they live as much as they know where you live. You probably have talked about work and employment, and hobbies and children. You may even know where they came from and how they came to your neighborhood. Your implied trust is that they are who they say they are. But the reality is, how much of this information have you verified? Do you trust your neighbor well enough to give them the access to your house when you are away?

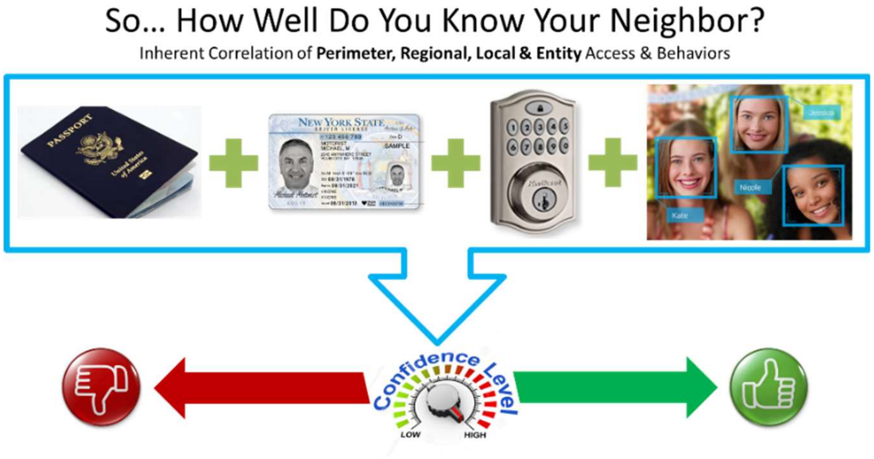

So… How Well Do You Know Your Neighbor?
Inherent Correlation of Perimeter, Regional, Local & Entity Access & Behaviors
PASSPORT
NEW YORK STATE
Jessica
Nicole
Kate
Confidence Level
LOW
HIGH

Figure 1: Introducing Zero Trust into an Enterprise Infrastructure, Checco (2018)

Your assumption is that their identity has been confirmed by various other entities in the community: their passport has been checked at the border, their license has been validated by the state, their employer has verified their right to work here, and other neighbors have corroborated the same information you’ve attained. All this external identification should have been performed by known trusted authorities before you ever met your neighbor. Ironically, your final step – the facial or voice identification of the person every time you talk with them – is known as continuous verification.

Zero-Trust performs identity and verification in much the same way. Identification is performed in-depth at various levels prior to access, and entity verification is performed continuously throughout the lifetime of the asset being accessed.

Fire Prevention (Compartmentalization)

Fire Prevention Week in the U.S. takes place annually aligned to the date of the Great Chicago Fire, a conflagration which occurred October 8, 1871.[3] The mission of fire prevention week is to reduce the number of avoidable fires due to negligent fire safety practices. Fire prevention experts are educating citizens about **flow path** – the direction the fire will move and grow due to open doors and small drafts in the residence.

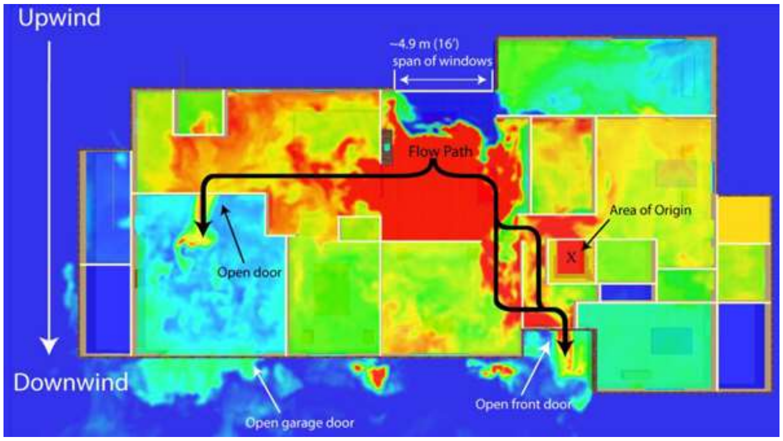
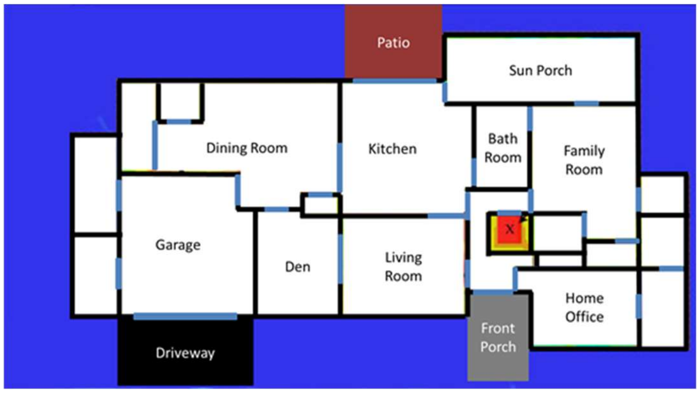


Figure 2: NFPA & NIST Fire Research Division

This education starts with live fire videos and infrared heat maps of where fire goes when all the inside doors are open. In this scenario, occupants have between 20 seconds and 2 minutes to vacate the residence safely. However, by simply closing doors the fire is contained to the room of origin, thereby choking the fire out and giving the occupants more time to react and find refuge.



Zero-Trust does the same thing to protect bad actors from spreading freely across an organization's resources. It cannot always prevent an attack, but it will contain the attack, minimizing its impact.

Finding the Right Balance (Orchestration)

Each concept above is in itself important and may not seem to be related, but the key to having great home security is to use them together in the right amount.

Throughout our neighborhood in the early 1970's, kids were free to roam and our doors were always unlocked during the day, we knew our neighbors, and everyone was welcome to share in our meals. But that all changed with the spike in serial killers during that decade (or at least the spike in news reporting and publicity).[4] With each news report came the realization that people needed to be aware of the potential risks, to: know where the kids were going, know who they were going with, keep doors closed during the day, and lock doors/windows at night.

Not every security measure that can be taken is the right method for every home. Each household has its own idea of what is **safe** – in security lingo, their **risk tolerance**.

Finding the perfect balance for a secure home space means knowing when to observe versus investigate information about the people you come into contact with, what type of locks and alarms to put on each door, and what your thresholds of personal space will determine an actionable event.

Zero-Trust Redux

Having been in the technology field since the birth of the personal computer and the internet, one question that always bothered me is: *Why did it take so long for ZT to get here?*

The term Zero Trust was coined at Forrester Research around 2010, but the concept of each asset being responsible for its own security (or trust) was a main tenet of early computing security with the mid-1980's popularization of the highly configurable operating system called Unix System V (in part due to the breakup of the Bell monopoly).

But … we technologists (collectively) were lazy with even the most rudimentary security controls when emerging and disruptive technologies were thrown at us.

A History of Disruption

Consider the following converging technology disruptions – from the 1980's through the 1990's – where a static set of intra-connected machines morphed into a massive dynamic inter-connected network:

- Token Ring networks and SNA protocol was the dominating flavor of connectivity, and Novell Networks was the leading provider for non-mainframe connectivity. This was quickly being usurped by Ethernet networks and IP protocols with both session-less (UDP) as well as session-based (TCP) overlays. There developed a propensity for building systems connecting both network types – network spanning – interconnecting mainframes to newer workstations.

- Sun Systems was disrupting the mainframe market with smaller more nimble computing power known as mini-servers and tightly coupled workstations, promoting Unix over the more complex MVS/CMS mainframe operating systems. To encourage an inter-connected mesh network model, Sun workstations and servers were shipped out-of-the-box in "trusted" mode to ease the assembly of enterprise networks. (SNA connectivity inherently uses an implied trust model relying on hub-and-spoke connectivity for devices.)

- Competition for personal computing workstation dominance was fast and furious, mostly between IBM and Microsoft. Whereas Microsoft Windows NT had introduced an embedded TCP/IP stack, IBM's OS/2 (which was essentially the same code base) was still pushing SNA, so TCP/IP was created as an "add-on" pack. As a result of poor integration, the low-level operations of IBM's TCP/IP stack for OS/2 was heavily reliant on user-editable environment variables without any authorization for overriding such environment variables. Windows NT was no better, though, as simple registry entry modifications could perform similar configuration changes at the user level.

- In 1981, as ARPANET migrated from academia to corporate sector and the introduction of HTTP/HTML with the Mosaic browser, inter-connected computing exploded. Routers replaced rudimentary Interface Message Processors (IMPs), and BGP and DNS were created to support the need for more intelligent connection routing as Terminal Interface Processors (TIPs) – aka endpoints – dramatically increased.

The practices of **network spanning**, **inherent trust, exposed configurability** and **commoditization of interconnectivity** created a toxic combination that still haunts us to this day.

Perimeter Security: One Size Fits All

During this period, the internet was an open forum and the need for self-governance was quickly becoming evident. Even though every Unix server had the capability to control access to the machine through YP/NIS, NSS and X.500 (the precursor to LDAP); firewalls gained popularity as a broad prophylactic measure to delineate the legacy intra-connectivity from the emerging inter-connectivity; the DMZ perimeter. But even then, default rulesets used an “Allow ALL” rather than a “Deny ALL” approach.

Providing more effective perimeter security has been around since the early 1990’s when the idea of layered security – originally used by military and law enforcement as a standard protection paradigm – was applied to technology and sometimes marketed as “Concentric Circles of Protection”, “Defense in Depth” and “Compartmentalization”[5]; but all are broad strokes of overlapping defenses.

With the proliferation of mobile devices, organizations were forced to manage yet another foundational technology vertical, and an entire subindustry around MDM (mobile device management) platforms arose. What this amounted to was yet another perimeter for encapsulating corporate access from a known unmanaged device.

What was Old is New Again

What was needed is a targeted approach to security that optimizes resources while providing just-in-time security measures at every stage (prevent, identify, detect, respond and recover); using the components and capabilities that have already been embedded into most Unix systems from the 1970’s.

As far back as 1994, the Jericho Forum promoted “de-perimeterization” – i.e. limiting implicit trust based on network location (intranets), the reliance on static single point of entry/exit and broad defense tactics over a network segment (DMZ). Many of the recommendations in this paper are extensions of – or tightly coupled to – the Jericho Forum Commandments[6] and the Global Identity Foundation’s “Identity 3.0” principles.[7]

Once again in 2001 – possibly resulting from the Y2K crisis – the promotion of individual security responsibility was re-introduced with IBM's "autonomic" computing architecture promoting systems that were self-configuring, self-healing, and self-optimizing. An interesting feature to IBM's hypothesis is that in addition to targeting individual systems, the autonomic paradigm could be employed as part of a set of tightly coupled systems, subnet or application stack … keep that notion in your back pocket.

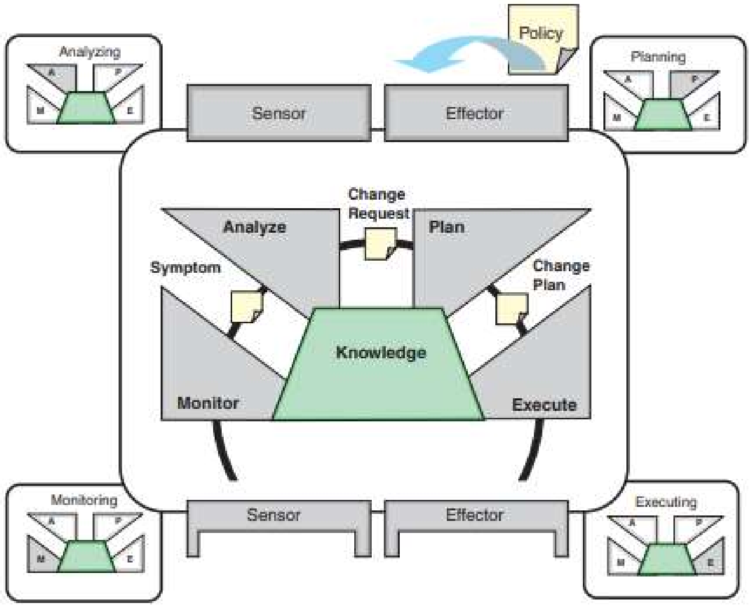


Figure 3: An Architectural Blueprint for Autonomic Computing - Third Edition, IBM (2005)

From IBM's Autonomic Computing architecture, we see parallels to the "Extensible Access Control Markup Language" (XACML) used by the Zero Trust Control Plane design:

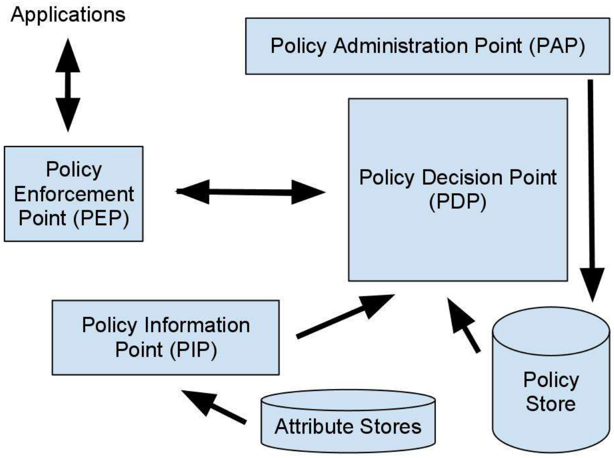

Applications
Policy Administration Point (PAP)
Policy Enforcement Point (PEP)
Policy Decision Point (PDP)
Policy Information Point (PIP)
Attribute Stores
Policy Store

Figure 4: XACML reference architecture

Functionally similar components can be mapped as follows:

- Sensors & Monitoring → Policy Information Point (PIP)
- Policy & Planning → Policy Administration Point (PAP)
- Analyzing & Change Requests → Policy Decision Point (PDP)
- Effectors & Execution → Policy Execution Point (PEP)

The main difference between these two systems is that IBM's Autonomic Computing model expects each resource to have its own sensor/PIP, analyzer/PDP and effector/PEP; whereas the Zero Trust paradigm optimizes these components into a single but distributed high-availability control plane.

The Zero Trust control plane model has one key advantage over the Autonomic Computing model: the control plane can gather, coalesce and analyze sensor data from a multitude of seemingly disparate sources to create an environmentally-aware analysis, deduce a context-sensitive decision, and ingest the feedback from that result stream into its machine-learning algorithm that will allow it to make better future analyses and decisions.

Why Zero-Trust

Even though the concept of Zero Trust is not new, now is the right time to rediscover this concept and implement a better security model.

Aspirational

It has been evident in the past decade that security threats are becoming more complex and persistent. From Firewalls to IDS to IPS/HIPS to WAFs to endpoint agents, each solution an organization purchases performs a progressively more complex operation to address a narrowly targeted set of attack surfaces.

A recent Ponemon-Devo study finds that 40% of those polled state their SOC has too many tools that overlap or produce redundant data. "*A new technology gets brought in and many of the older technologies [overlap] ... another thing gets added on the stack, and there's not thought on how to optimize them*," Julian Waits, general manager of cybersecurity at Devo.[8]

If one were to map security spend over time versus the attack surface protected, the Return on Security Investment (ROSI) is ever decreasing[9] and may eventually reach zero. But security is a **response model** rather than a production model; i.e. some spend must occur as brand/reputation and regulatory factors may require. The research paper “Identifying Unintended Harms of Cybersecurity Countermeasures” hypothesizes as more security measures are deployed, new unintended consequences are introduced, which then must also be addressed requiring even more countermeasures, proliferating the cybersecurity technology spread.[10]

The Zero-Trust principles allow an organization to reimagine their security in a way that does not require incremental spending in the long run. However, the initial spend is quite high and requires a huge amount of resources to map assets, analyze existing entitlements, and refactor roles and access controls. Thus, an actual implementation of Zero-Trust would truly be an aspirational goal.

Google, a corporate giant known for its forward-looking culture, created a ZT reference implementation known as BeyondCorp from scratch. All BeyondCorp applications are cloud-native and embed zero-trust principles across entity directories, applications, routing and data elements. The reality is that not all of Google is ZT, only the small subset of users and applications in the BeyondCorp domain.

Although a reference implementation, Google demonstrated that ZT is feasible; it will just take an omniscient outlook, a clear focus on a targeted scope, lots of resources and multi-year commitments.

Overdue

The 2020 pandemic crisis has forced the acceleration of an all-remote workforce, and this unplanned initiative has stressed existing installations of VPN connectivity and exposed many security gaps with legacy infrastructures. As a result, there has been a renewed interest – perhaps even a frenzied approach – to accelerating the movement of business to the cloud.

"*Moving to the cloud*" is more than simply becoming a tenant to a third-party platform/infrastructure, it also encompasses the *virtualization* and *containerization* of systems running on those cloud instances and requires new skills for managing the orchestration of instances. Cloud migration can be accomplished in three ways (or any combination thereof):

- **Replacing with SaaS** is a low friction migration where third party applications replace existing systems with an out-of-the-box cloud-based solution. This is best for those functions that can be commoditized, such as HR, CRM, benefits, payroll and logistics; but are not well suited to migrate sector-specific or homegrown applications. Moving to a new platform altogether can introduce new vulnerabilities and weaknesses; and your organization is at the mercy of the vendor to respond and recover in a timely fashion.

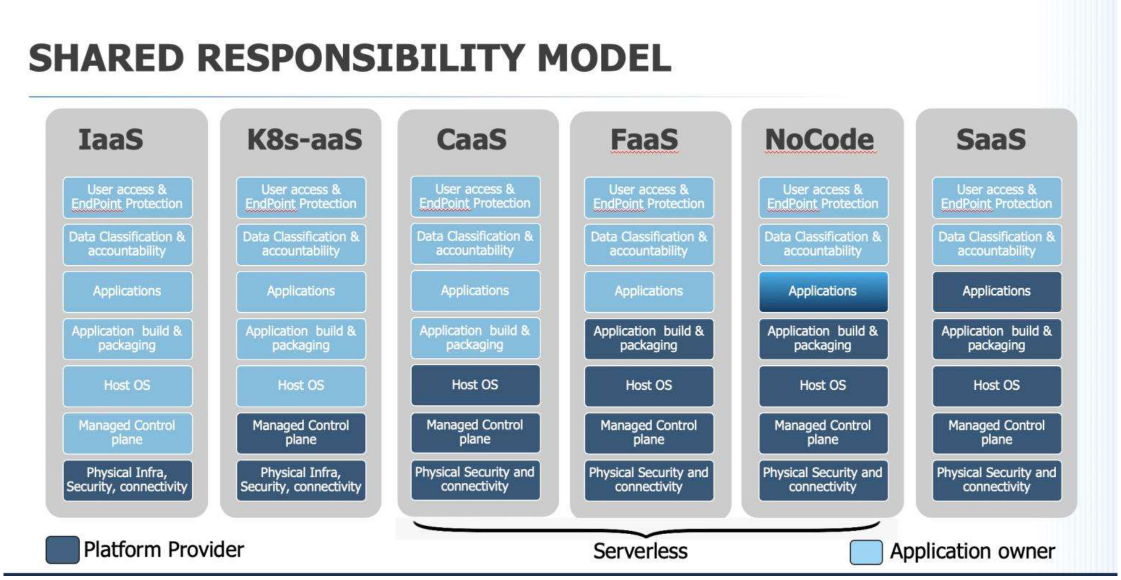
- **Uplifting to IaaS** is the refactoring of a legacy system from a physical server to a cloud-based infrastructure. This can vary in friction based on their coupling to other upstream and downstream systems. The least friction is seen with systems leveraging modular components and APIs; the most friction comes from systems built using customized communication and exchange protocols. Also, note that any known weaknesses in a system are exacerbated when moving to the cloud; and new vulnerabilities will be discovered when the shelter of perimeter-based security measures is removed. Mitigating such weaknesses will be difficult as there may be lack of knowledge transfer or SMEs, lack of resources to address the issues, or the inability to resolve due to flaws in the original system design.

- **Rebuilding in PaaS** is the most expensive, but best long-term solution for moving a business process to the cloud. It takes a business process and implements it as a set of cloud-native and containerized systems using the latest technologies and security designs. Rebuilding an application to be cloud-native can also involve changing the operating paradigm from recoding to PaaS to more serverless options such as Container-as-a-Service (CaaS) or Function-as-a-Service (FaaS). However, rebuilding is resource intensive, requires the re-thinking of the process from end-to-end, and is expensive. As in the other scenarios, rebuilding a system from scratch will introduce vulnerabilities and weaknesses; yet, there is the advantage of being able to respond and recover quickly.

Cloud removes the need for bursting VPN capacities, and provides a consistent set of ingress and egress paths for users and data and virtualization/containerization allows for the ephemerality of systems in the event of rogue access or operational instability.

Access-Anywhere & Security

What cloud does not inherently provide is any level of *security*; not at the perimeter (as there is no perimeter) and not for individual resources. Security is the responsibility of the tenant, and CSA's cloud controls matrix (CCM) provides a standard set of security controls that should be implemented by the tenant. But standardized cloud security tactics do not exist, cloud providers are inconsistent in the level of security tools they provide tenants. In addition, the level of security responsibilities change depending on the cloud model that is adopted for each application; so, an organization could be responsible for different levels of security in a single tenancy with multiple cloud-based applications.



SHARED RESPONSIBILITY MODEL

IaaS	K8s-aaS	CaaS	FaaS	NoCode	SaaS
□ User access & EndPoint Protection	□ User access & EndPoint Protection	□ User access & EndPoint Protection	□ User access & EndPoint Protection	□ User access & EndPoint Protection	□ User access & EndPoint Protection
□ Data Classification & accountability	□ Data Classification & accountability	□ Data Classification & accountability	□ Data Classification & accountability	□ Data Classification & accountability	□ Data Classification & accountability
□ Applications	□ Applications	□ Applications	□ Applications	■ Applications	■ Applications
□ Application build & packaging	□ Application build & packaging	□ Application build & packaging	■ Application build & packaging	■ Application build & packaging	■ Application build & packaging
□ Host OS	□ Host OS	■ Host OS	■ Host OS	■ Host OS	■ Host OS
□ Managed Control plane	■ Managed Control plane	■ Managed Control plane	■ Managed Control plane	■ Managed Control plane	■ Managed Control plane
■ Physical Infra, Security, connectivity	■ Physical Infra, Security, connectivity	■ Physical Security and connectivity	■ Physical Security and connectivity	■ Physical Security and connectivity	■ Physical Security and connectivity

■ Platform Provider

Serverless

□ Application owner

Figure 5: Vishwas Manrel, NanoSec

Zero-Trust works best with cloud / containerized applications, as it takes advantage of the dynamic access capabilities, and orchestrates individual user activity with individual resource activity, ensuring both gross authentication and acute entitlements. With such fine-grained granularity of control, any breach of an asset (which will inevitably happen) should be limited to that one asset and impact should be minimized.

Resource Elasticity: Moving Pets to Cattle

Cloud implementations also introduce the concept of *resource elasticity*, where multiple instances of the same system can be instantiated or destroyed as demand requires. If systems are not designed specifically to handle such dynamic capabilities, the business may end up having orphaned transactions, reporting inaccuracies, or full-stack system failures.

Zero-Trust, however, does not solve the elasticity problem; and may in some cases intensify the issue of elasticity if the organization embraces the D.I.E. (cattle) vs C.I.A. (pets) methodology[11] without properly refactoring systems for ephemerality.

Dynamic Networks

Orchestrating systems to run a cloud (or container) environment solves just one part of the problem; the interconnectivity of systems which make a process flow – and which protects a process from unauthorized entities – is the responsibility of proper configuration and access controls.

Organizations have traditionally used perimeter security to delineate external (untrusted) from internal (implied trust) access; it is also repurposed for defining internal subnets that create boundaries for system access to specific systems and users based on functional roles. The issue becomes that as organizations grow and morph, role definitions become muddy and internal protection mechanism handle more exceptions than rules.

Zero-Trust obsoletes the idea of a statically defined network. Connectivity between assets is not a physically routed path, but rather a dynamic access decision based on a wide swath of sensor data. This is known as software-defined networking (SDN) and network function virtualization (NFV).

Not Everything is ZT-Capable

It is important to remember that not every system can be virtualized, migrate to the cloud, provide or ingest sensor data, allow fine-grained entitlement access or become cattle. Some critical systems need to be tightly coupled to an internally protected network, some systems are too legacy to refactor, and some systems may have specialized hardware needs that a commoditized infrastructure cannot support.

Our recommended approach to such situations is to apply the Zero-Trust concepts to group of a functionally-related tightly-coupled legacy systems – i.e. creating as small a perimeter as possible and treating that grouped subsystem as an individual asset. This notion dovetails into a larger more comprehensive Zero-Trust layered approach talked about later in this document.

Defining Zero-Trust

Many pundits have proclaimed that Zero Trust is the death knell for perimeter-based security measures; and Google's reference implementation known as BeyondCorp exemplifies how a purist model of Zero Trust could be accomplished. The caveat to BeyondCorp, though, is that it is only a small slice of all applications and users in Google, it only deals with web-based applications, and applications were designed and built from the ground up to be ZT-aware.

The reality is that any enterprise ZT strategy must embrace the existing measures, not break them. As seasoned practitioners, we must promote the idea of **holistic change**, not wholesale change. This means that, for now, perimeter and other legacy security solutions remain in place. If one were to adopt the "rip & replace" approach, we are left to re-invent the wheel, to reconsider all aspects of existing security measures, and struggle to adapt new paradigms to legacy infrastructure.

Zero Trust capabilities must be built independent of – and complementary to – existing solutions. It is better to have overlapping security rather than missing, or even worse, conflicting security controls.

Prerequisites

The holistic approach to a Zero Trust paradigm has some significant prerequisites before the first tactics, feasibility planning or proof-of-concepts can be executed.

Since 2018, NIST's National Cybersecurity Center for Excellence (NCCoE) has hosted a Technical Exchange Meeting focused on Zero Trust discussing the NIST SP800-207 drafts and current implementations. What became evidently clear is that each participant – a mix of government agencies and outside invited private enterprises – had very different interpretations of what Zero Trust meant from an implementation perspective.

Whereas one agency had decided that Zero Trust was solely an IAM solution (PIV cards, smart cards, tokens, smart badges), another agency embraced a purely micro-segmentation implementation (and Kerberos certificate-based authorization). Neither implementation was close to complete but secured enough of their infrastructure to "check the box" on their ZT objective; but clearly there are gaps between those two extreme interpretations. Then there was an agency that was attempting to implement policy enforcement at every step in the request – a purist view but technically infeasible.

DISA recognized the need to coordinate an overall common ZT definition across departments to prevent security gaps. This epiphany became part of the NIST SP800-207 draft update, although not as extensive as covered below.

Common Lexicon

To communicate effectively every party needs to speak the same language, from leaders to management to operators to engineers to vendors. Having a common lexicon is important. A term or acronym can result in vastly different reactions depending on the audience.

This cannot be more evident than the use of similar terms between technology and military personnel; where misinterpretations were prevalent with terms such as: DMZ, firewall, security, trust, confidence level, strategy, architecture, framework, network and policy.

For example, note that using the term "infrastructure" in this document may seem counter-intuitive since a pure Zero-Trust implementation would have no perimeter, no boundaries, and thus not considered an infrastructure. Zero-Trust, in contrast, has "flow paths" that encompass autonomous requests and responses across open networks to reach an authorized resource. For all intents and purposes, we shall use the term "infrastructure" to represent all the valid flow paths within an organization.

It is imperative to create a baseline of understanding that can be shared prior to meeting with any groups in your organization (or with any vendor) to ensure clarity of communication.

Asset is the Perimeter … What's an Asset?

Beyond the accepted postulation that **data** is an asset that needs protection, many pundits recite the Zero-Trust mantras of "the user is the perimeter" or "data is the perimeter" … and some vendors have taken the liberty to spin that into "the endpoint is the perimeter" as it aligns better to their product offerings.

A pragmatic look at those approaches reveal serious flaws. If the user was truly the perimeter, they would indeed operate in a protective bubble, but that would not protect the leakage of PII and NPI stored by third party systems; it would only protect an individual's access to that data. This only works if two main assumptions can be proven: (1) all individuals are perimeterized, and (2) systems holding that data never fail.

Self-Driving Vehicles is an excellent allegory to demonstrate this concept. We have seen that accidents will still happen when roads are made up of both self-driving vehicles and human-driven vehicles, because human decision making and AI-based decision making are different. One might surmise that solving this problem requires all decision making to be the same. Once all vehicles are self-driving – all decision making uses the same predictable algorithm – then there is no chance for misinterpretation of intention.[1] What we have then failed to account for, though, is the assumption that all road conditions are clear and observable. The reality is, our roadways – the infrastructure of transportation – has lots of variations and much is in disrepair.

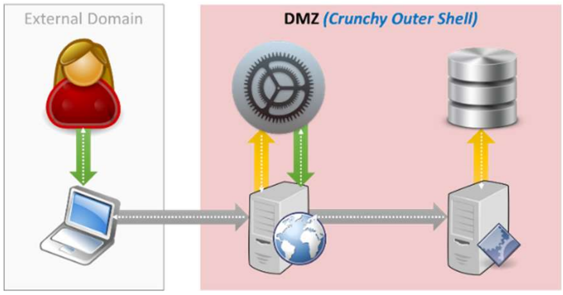
Similarly, focusing on "user is the perimeter" requires that ALL users (innocent and nefarious) abide by the same rules and that systems holding NPI never fail. How much confidence does your organization have both those requirements can be attained and executed with perfection? In your organization's infrastructure, what is the state of patches on existing systems?

Finally, if organizations attempt to treat data as the perimeter, they'd quickly find that the volume of metadata generated would quickly become untenable for storing, synchronization, validating, monitoring and tracking. Metadata would quickly become stale, inaccurate and orphaned; adversely affecting the systems using that metadata for protection.

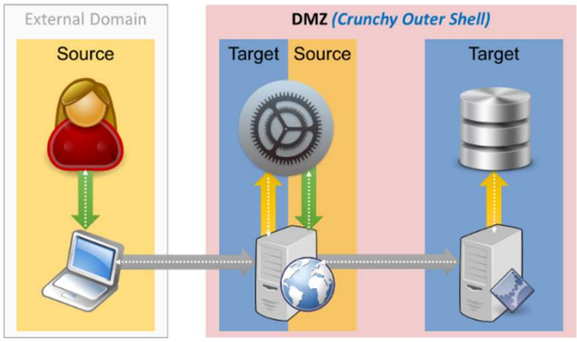
True protection of users and their data requires more than just focusing on just the user or just the data; Zero-Trust means providing fortifications around every asset that a user or their data can be accessed from.

In the real-world implementation of Zero Trust, the ***Asset is the Perimeter*** *.*[12]

To fully comprehend the Zero-Trust definition of an asset, let's look at the typical flow path where a user retrieves a piece of information:

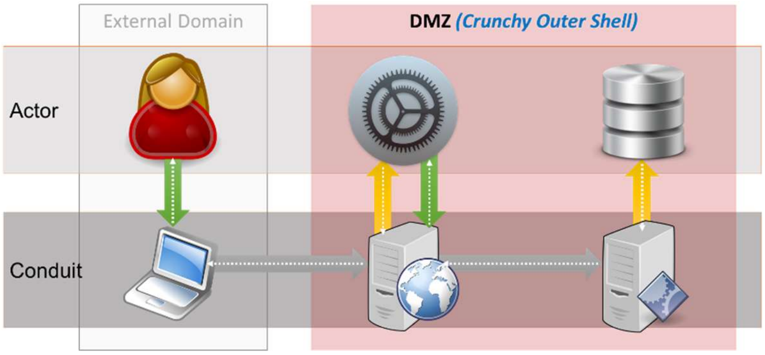


Our first pass simply defines what is a "source" and what is a "target" for any information request:



This delineation, although quite obvious, plays an important role in scoping attributes to be used for the Zero Trust control plane.

Next, we need to differentiate the components which store and use information (the "actor") versus those which simply transport the information (the "conduit"):



This non-obvious distinction is necessary to understand how policies are enforced, and where gaps may exist between the **asset of vulnerability** versus the **asset of enforcement**.

Thus, the Zero Trust view of information flow disregards the perimeter, leaving an **asset-centric flow** that looks like:

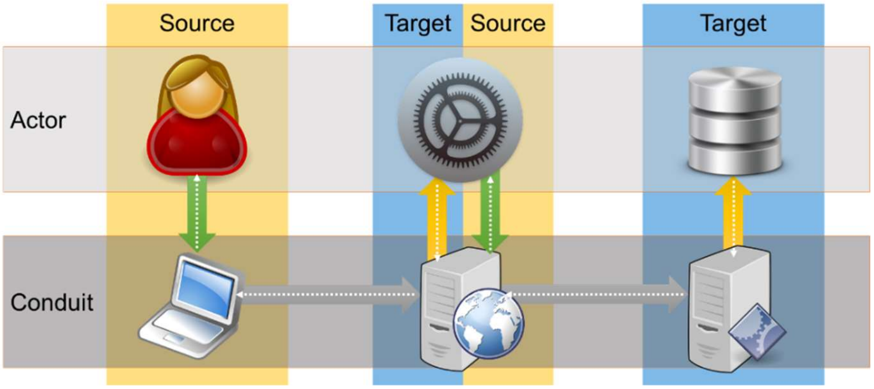


Figure 6: Introducing Zero Trust into an Enterprise Infrastructure, Checco (2018)

It is imprudent to think that simply protecting the asset itself is sufficient. To be fully covered, there is the need for the protection of **workloads** (transactions in flight and/or information in transit) – which, is the **coordinated responsibility** of both the sending and receiving assets. In this above simple example scenario, Zero Trust must address the protection of six assets, as well as the protection of information across those assets, which would be six additional coordinated transports. Identity 3.0 principles confirm that "*risk will probably be bi-directional and* ***both entities in a transaction will share the risk****, though usually disproportionally*."[13]

Having the correct frame on a problem is the first of six elements in Carl Spetzler's **Decision Quality** (DQ) success chain. Building a functionally complete Zero Trust strategy must account for all types of assets and ensure coordinated protection across all facets.

Congruence of Strategy & Direction

Keeping in step with the idea of Decision Quality, the DQ success chain has as its last element, "*Commitment to Action*" which obviously refers to the execution of any planned changes.

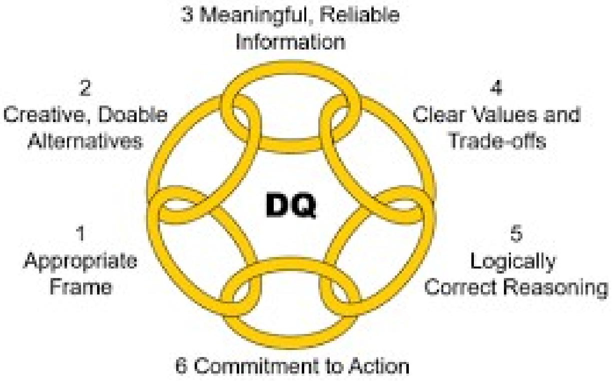

3 Meaningful, Reliable Information
2 Creative, Doable Alternatives
4 Clear Values and Trade-offs
DQ
1 Appropriate Frame
5 Logically Correct Reasoning
6 Commitment to Action

Figure 7: Strategic Decisions Group

However, before one can assure that commitment, even before the first stage of problem framing, the DQ process states there must be a concerted effort to bring all the affected parties together to agree on two important items: "Recognize the Situation" and "Agree that a Decision" must be made.

In a project as large as Zero-Trust, we recommend that the participants first get training in the Dialogue Decision Process, as many alternatives and trade-offs must be decided on.

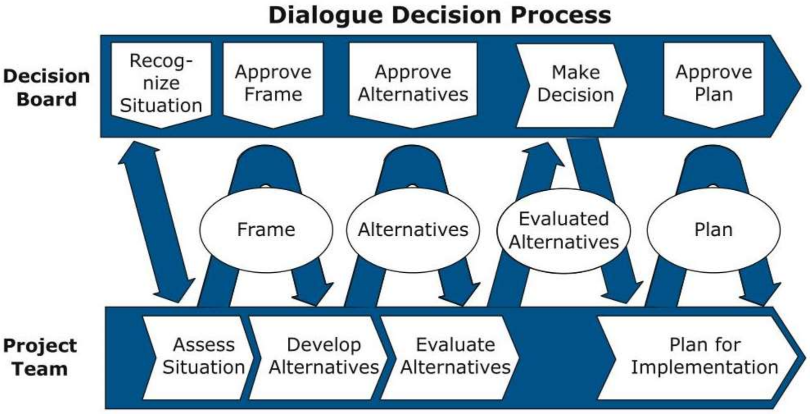


Figure 8: Strategic Decisions Group

The Dialogue Decision Process disconnects decision-makers from decision-analysts, allowing for a more measurable, documented and defendable outcome, with less political influence.

Standard for Information & Data Exchange

The magic of Zero-Trust's dynamic decision-making process is partly due to the ability to gather, normalize and correlate log data from various semi-related "sensors" across the request flow.

Security operations has greatly benefited from standards such as MITRE's ATT&CK framework[14], Structured Threat Information eXpression (STIX)[15] and Trusted Automated Exchange of Intelligence Information (TAXII)[16]; but these are aligned to tracking known IOCs and TTPs.

What Zero-Trust needs is a standard data exchange format and information mapping that allows devices of all types can share flow path details in normalized manner.

Tactically, this requires alignment across solutions in a market where every vendor is attempting to be the core Zero-Trust provider; thus, each vendor sees themselves as the de-facto standard for data ingestion and orchestration.

Orchestration Platform

Dovetailing on the standardization issue is the need for a true Zero-Trust orchestration platform; one that supports the ingestion of data from various brands of flow path sensors and asset agents, can administer policies for a robust variety of domains, act as a machine learning decision engine, and provide ubiquitous enforcement of policies across the organization's infrastructure.

Why is orchestration so important? One could argue that "cloud" is simply is a set of virtualized computing resources. But what makes it "the cloud" … orchestration. Orchestration is the ability to simplify the management of many interconnected resources into a cohesive and usable interface. For the cloud, orchestrating elasticity and configuration of computing resources were the functions needed to commoditize virtualization.

> *What orchestration did for transforming virtualization into the Cloud,*
> ***orchestration** will transform security controls into Zero-Trust.*

Orchestration, to some degree, can be accomplished today. With your existing security tools, determine what sensor data they can provide, categorize those sensors by their asset coverage into policies, as in the example below. The challenges though are twofold: (1) extracting and making efficient use of sensor data with a decision engine, and (2) be able to identify the security gaps against their ZT target state.

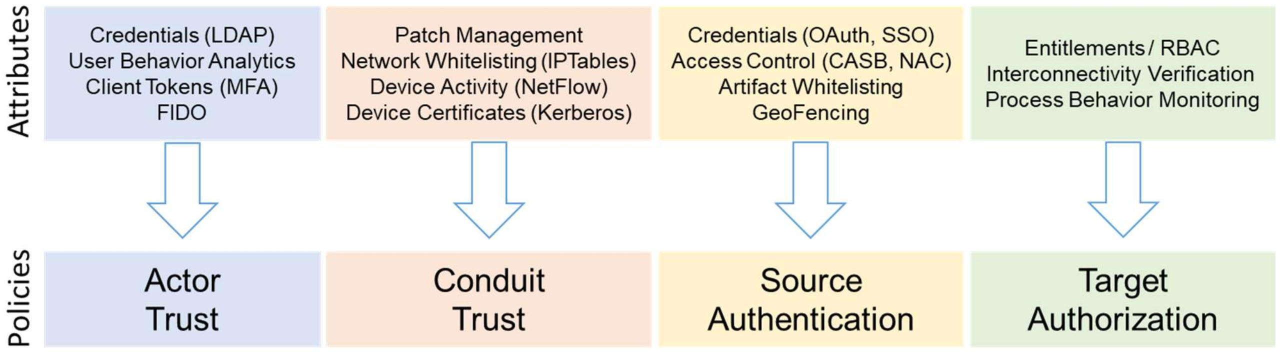


Figure 9: Introducing Zero Trust into an Enterprise Infrastructure, Checco (2018)

Within the next decade, we surmise there will be consolidation amongst vendors; but not before there is widespread divergence of data formats, proprietary APIs and narrow-banded functionality. At the end of this immaturity will be two or three major independent control plane implementations, which security vendors will integrate to. But until that cooperative future exists, organizations are left to create data transformations themselves – which implies that they must select a strategic vendor to use as their orchestration platform and conform to that data exchange format.

Foundational Components

In general, envisioning a Zero-Trust Architecture (ZTA) requires several key components: asset inventory, continuous verification, compartmentalization, independent control plane, and playbooks for handling indirect affectations. The majority of these components can be initiated using current infrastructures and migrate to a ZT target state.

Asset Inventory

A key principle in Zero-Trust is having visibility into all the organization's assets, regardless of how they may be protected. Unfortunately, this is an age-old challenge is so daunting that it has spawned its own subsegment of vendors focusing on just capturing technology inventory. And even with each vendor's success in this market, there still exist unabating issues around rogue devices, orphaned identities and spot solutions.

The Hunt for Red October (Rogue Devices)

Over time, networks grow in size and scale rather than shrink. As new systems are introduced at an ever-increasing pace, there is the potential for devices to be inserted without detection or management: rogue devices. When it comes to understanding rogue devices, one must consider various scenarios:

- **Unmanaged Networked Devices**: These devices cannot be managed using enterprise remote management interface (RMI) tools as they cannot support the RMI agent, inadvertently deployed without the RMI agent, or the RMI agent does not respond to inquiries due to misconfiguration or port blocking.
  - Such devices, if known, must be inventoried, patched and managed manually.
- **Endpoint Peripherals**: includes personal devices on the network as well as unknown peripherals plugged into existing managed corporate devices via USB, Bluetooth or Lightning connections.
  - Although it would be easiest to totally disable peripheral connectivity, usability would then be severely compromised including the ability to use a USB-based keyboard, mouse or USB-C display. But allowing those specific peripheral types opens the door to keyloggers and data capture devices mimicking the valid exceptions.
  - One solution an organization implemented was to have manufacturers for specific devices create a batch of devices using a special organizational vendor-ID; thus, all corporate endpoints blocked all peripherals except for that vendor-ID.
- **Shadow IT**: includes servers or cloud-based workloads performing unsanctioned business functions. This can be manifested as desktops running server software, development/test systems running against a production database, servers running in office environments (rather than a datacenter), or personal cloud instances exposing business applications to the internet.
  - To identify Shadow IT, many organizations have instituted automated triggers on corporate credit card activity referencing cloud service providers and engaged external solutions that scour the internet for all branded activities.
  - Cloud-based "no-code" initiatives, such as AWS Honeycode, make rogue applications/workloads even more difficult to detect.
- **Unauthorized Routing**: refers to devices on the network used by threat actors for remote access or data leakage, such as a Pineapple or a split-tunneling access point.
  - These devices are characteristically difficult to pinpoint as they are built to mimic existing network behavior.

The marketplace for finding rogue cloud instances, devices and peripherals is fairly mature. Unfortunately, the weaknesses lie within an organization's lack of appetite for blocking unknown network/server activity – fearing it would break undocumented obscure business processes – as well as overly broad safelists of peripherals.

Finding Nemo (Orphaned Accounts & Processes)

Another area of concern when preparing for a Zero-Trust implementation is the search for orphaned identities – those accounts remaining after someone has left the organization. This involves mostly systems that are missing from the asset inventory, vendor-hosted, or otherwise unmanaged.

Orphaned accounts are not a new problem, yet we have breaches exploiting inactive accounts spanning from LendingTree in 2008 to Quora in 2019, the most egregious against RSA in 2011.

Not all orphaned accounts are vulnerable oversights … some are existing "service-IDs" used by internal services that were originally owned/created by the former employee. However, once the owner has left the organization, there offboarding process has failed to re-assign ownership of the credential or ensure knowledge transfer of the running service. In some cases, the orphaned services have been running through multiple *generations* of employees, so any historical knowledge of the business logic (and sometimes even source code) has vanished.

Orphaned accounts and processes clearly become a concern for migrating legacy processes to a Zero-Trust model. The tactics for identifying orphaned accounts needs to encompass both a top-down approach as well as a bottom-up approach. The top-down approach is the auditing of an organization's account system which includes: the centralized LDAP, PAM lists on all servers, and local credential datastores for all applications. The bottom-up approach utilizes both existing WAFs and API tracing tools to trace all internal or outbound network traffic that requires authentication such as WSSAPI, SAML, OAuth, and TLS inspection via centralized termination points or proxies. These multiple sets of data points then need to be reconciled against a known active account baseline, and predetermined remediation plans should be executed as feasible – i.e. re-assign ownership, observe-and-report, block, or remove.

Be warned, dealing with orphaned accounts will be much more complicated and fluid than expected.

Tower of Hanoi (not the Tower of Babel)

Although our proposed definition of assets uses functionally categorized semantics such **as actors, conduits, data** and **workloads**; the marketplace is still divided into more physical definitions: users/identities, managed/unmanaged devices, networking resources, and cloud tenancies.

Regardless of the capabilities of existing solutions to capture the full breadth of assets in an organization's infrastructure; the inability of these vendors to manage, share or correlate data into a common repository remains an open problem.

For a Zero-Trust architecture to be effective,
data from disparate sources must be ***harmonized****.*

If current vendors cannot integrate or provide congruent data, then the usability of 100% visibility into resources will be stunted.

Continuous Verification

To be clear and distinct from all those “zero-trust” vendors: The User is NOT the Perimeter! This does not however exclude users from the zero-trust equation; rather users are just one type of “actor” asset to be protected.

Zero Trust requires a mature identity and access management program. It is more than just purchasing a popular IAM technology, even if it claims to be a Zero Trust solution. As a security architect, an **IAM program** must take into consideration detailed requirements covering both Authentication and Authorization.

Authentication (AuthN): “Who Are You?”

Authentication is the precursor to many zero-trust concepts; as it provides two main functions: identification and verification.

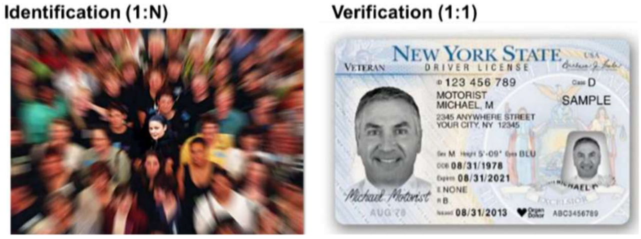


Figure 10: System Design for Biometrics, Checco (2017)

- <u>**Identification**</u> **is a 1:N relationship.** Identification is the selection of a single individual amongst a population of potential users. Ideally, the identification process should result in a single user record, but selection can be time-consuming.

- <u>**Verification**</u> **is a 1:1 relationship.** Verification is the process of tangibly confirming that a selected individual is who they say they are, given a keyed piece of private data.

Traditional credentials such as user-ID / password combinations easily satisfy both identification by limiting the search population indexed by the user-ID and verification by matching the password to that selected user record.

However, non-linear identification methodologies (facial recognition and other biometric technologies) can make the identification process more complex and less accurate, resulting in a range of possible matches. Biometrics are better suited for verification. But behavioral (and some physical) biometrics return verification as a confidence measurement as opposed to a binary decision; thus, it is up to the decision target asset to make the final decision based on some static or dynamic threshold.

Zero-Trust proliferates the need to manage ***non-binary decisions****.*

The control plane policy decision point (PDP) will create a confidence measurement based on both binary and non-binary information garnered from multiple resources. For example, although a traditional uid/pwd may be a binary result, but the geolocation of the endpoint they are authenticating from may be questionable; therefore, requiring a more context-based decision by the targeted asset.

Zero-Trust benefits from a ***FIDO2****-based authentication model.*

The advent of FIDO2, where identities are biometrically verified by the endpoint device, has greatly offloaded much of the authentication processing resulting in a more risk-aligned security model, faster access, reduced liabilities, improved privacy and a more seamless user experience.

Authorization (AuthZ): "What Can You Do?"

Authorization is like a museum guide; it determines where an identity goes and what information it can use. AuthZ assumes authentication has taken place and provides two basic functions: access and entitlement.

Figure 11: System Design for Biometrics, Checco (2017)

- **<u>Access</u> is a coarse-grained barrier.** Access simply guides entities where they can and cannot go. When access rules follow an "Allow All" paradigm, entities can traverse the network anywhere except where blocked from specific targets. In a "Deny All" mindset, entities can only access specific targets. As we have mentioned earlier in the history behind Zero-Trust, many networks were defaulted to "Allow All" and Zero-Trust conveys a "Deny All" approach.

- **<u>Entitlement</u> is a fine-grained barrier**. Entitlement determines where and entity can navigate to, what data the entity can see, and limit what actions the entity can perform.

Zero-Trust will require a revisit to role-based access control.

Zero-Trust requires a well-run role-based access controls (RBAC) operation, although most organizations have implemented minimally viable RBAC. Building a robust RBAC deployment is difficult because access and entitlement rules are quite different animals:

	Granularity	SME	Assignment Complexity	Repository	Execution	Management
Access Rules	Coarse (Allow/Deny)	Business Manager	Per: … user/role … application	External (LDAP)	WAF, CASB or LDAP	Linear
Entitlement Rules	Fine-Grained	Application Developer	Per: … user/role … application … operation … data element	Internal (Hardcoded)	Application Logic	Exponential

Defining coarse-grained access rules is relatively straightforward.
*Detailing **fine-grained entitlements** is arduous and complicated.*

Zero-Trust will require that both application-based access and entitlements be defined, documented and stored in a central accessible location. A greenfield infrastructure entrenches this as part of the software development lifecycle (SDLC), the data lifecycle management (DLM) and the continuous implementation / continuous deployment (CI/CD) model. In the brownfield legacy infrastructures, legacy applications need to be unwound and documented; and even then, may require thoughtful exception-handling as the existing code is unalterable in its current form.

Mutually Assured Authentication

IAM not only encompasses a people-centric security approach but extends to all types of assets (actors and conduits). Ideally, users authenticate to applications (the actors) whereby and endpoint authenticates to a network and eventually has access to the target server (the conduits).

*Zero Trust mandates **mutual verification** amongst connecting entities.*

In the current scenarios, **connectivity measures provide access but not verification**. Whereby the VPN router verifies the requesting endpoint and the application verifies the requesting user, there are no reciprocal measures. The endpoint doesn't verify the network's identity nor the application server's identity.

The lack of mutually assured authentication is a key shortcoming for many IAM solutions in the market, as they only deal with carbon-based endpoints. To address mutual connectivity verification, solutions may use one or more of the following methods:

- **Kerberos**[17]: is a protocol developed at MIT and released to the public in the 1980's which uses strong cryptography to allow a client to prove its identity to a server (and vice versa) as well as subsequent communications between the two. Although still used by many high-tech organizations, it has failed to reach popularity due to its management complexity and usability.

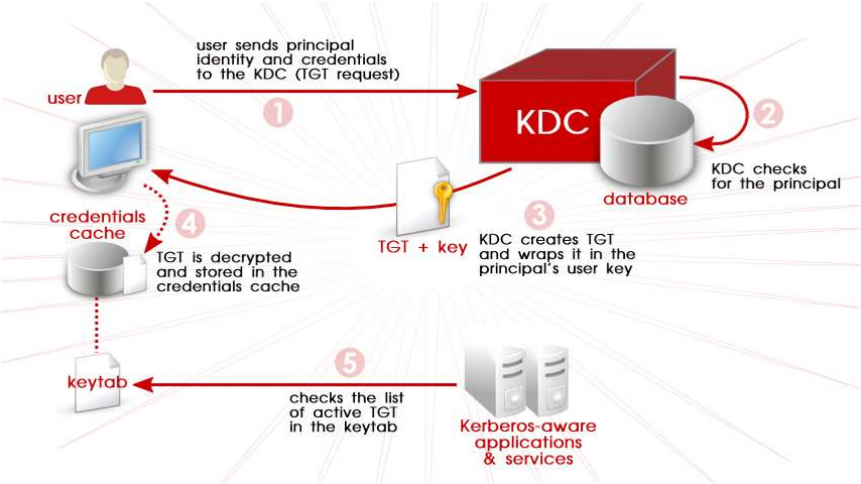

user sends principal identity and credentials to the KDC (TGT request)
1
user
KDC
2
KDC checks for the principal
database
credentials cache
4
TGT is decrypted and stored in the credentials cache
TGT + key
3
KDC creates TGT and wraps it in the principal's user key
5
keytab
checks the list of active TGT in the keytab
Kerberos-aware applications & services

Figure 12: Using Kerberos Red Hat Enterprise Linux 6 (Red Hat Customer Portal)

- **Single Packet Authorization (SPA)** [18]**:** uses UDP packets for TCP session pre-authentication, a concept which has been in use for over 10 years in SSH2/SCP2 and TLS protocols. This is an effective method for addressing network resources, but not other types of assets.

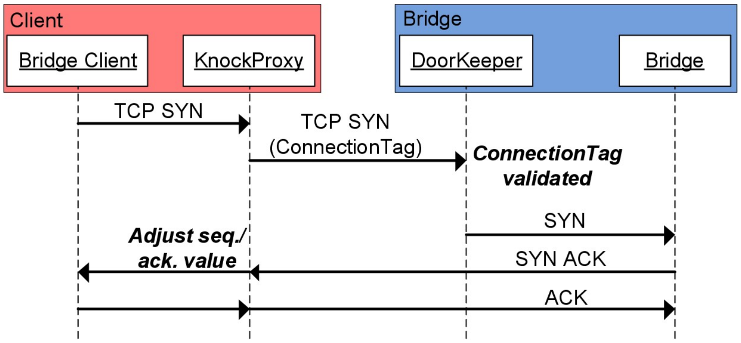


Figure 13: BridgeSPA: A Single Packet Authorization System for Tor Bridges

- **Secure Production Identity Framework for Everyone (SPIFFE)**[19]: defines itself as "open-source standards for securely identifying software systems in dynamic and heterogeneous environments." SPIFFE has shown much promise in that it focuses on securing *workloads* rather than data, systems or resources; however, commercial implementations to date have been less than ideal in both functionality and scalability.

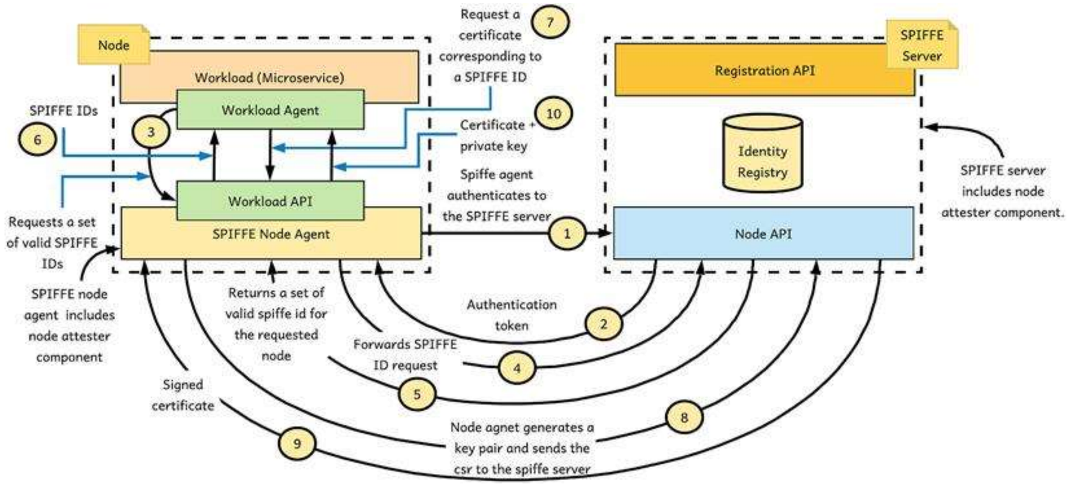

Node
Workload (Microservice)
Workload Agent
Workload API
SPIFFE Node Agent
SPIFFE IDs
6
3
Requests a set of valid SPIFFE IDs
SPIFFE node agent includes node attester component
Returns a set of valid spiffe id for the requested node
Signed certificate
9
Request a certificate corresponding to a SPIFFE ID
7
Certificate + private key
10
Spiffe agent authenticates to the SPIFFE server
1
Authentication token
2
Forwards SPIFFE ID request
4
5
Node agnet generates a key pair and sends the csr to the spiffe server
8
SPIFFE Server
Registration API
Identity Registry
Node API
SPIFFE server includes node attester component.

Figure 14: Microservices Security in Action MEAP V08 (Manning)

All three methods above have their advantages and disadvantages, as they each focus on a different perspective of access assurance between entities. It is recommended that organizations not rely solely on a single method for mutual verification of (at least) critical assets; rather, utilize multiple techniques – in an orchestrated fashion – to provide overlapping assurances as well as security resiliency.

Know Your Connector (KYC)

Where financial regulations require KYC (Know-Your-Client) assurance for every transaction, Zero-Trust requires similar KYC assurance between all connected parties. This involves continuous bidirectional verification, not only of static identity but also of dynamic behavioral activity. Known as UEBA or User/Environment Behavioral Analysis, humans perform this innately (with varying levels of success).

UEBA solutions in the marketplace are purpose-built to track carbon-based or silicon-based entities; and the basic premise for anomaly detection is to build a baseline behavioral map and track the delta or variance from the norm over time. Anomaly detection solutions all have the same inherent challenge: how accurate of a clean baseline can be made from a dirty network or a corrupt individual? The answer is simple: baselines should reflect the expected behaviors of the roles and functional responsibilities of the entity being mapped. Implementing that answer is complicated, as it requires byte-level knowledge of all valid decision paths in any business logic flow.

Even if an application has documentation, it is highly unlikely that all types of discourse were considered or documented. Even the designers/developers of those legacy systems cannot map all the proper logic flows with the correct data elements to be used and the *proper order* they need to occur. Early UEBA solutions relied on obtuse reverse engineering of application interfaces (of which they have no operational knowledge of) or harmonizing activity over time to determine most likely flow paths – all of which results in False-Rejection-Rate (FRR) triggers when non-routine actions occurs (i.e. exception handling), or False-Acceptance-Rate (FAR) indifference for actions that produce a toxic combination effect (i.e. sidestepping "segregation of duties" controls for fraudulent actions). Newer UEBA solutions can more accurately deduce expected logic flows from aggregate activity across an application, and others include sentiment analysis of the user based on prior activity; but many still cannot connect the contextual conditions surrounding anomalous action to determine *motives* for the activity.

Compartmentalization

Network complexities increase exponentially over time as more infrastructure is layered over older ones for backward compatibility. Having a network with resources that operate under different rules opens the door for vulnerable gaps to be exploited. Compartmentalization is a methodology used to ensure operating paradigms are not crossed is to segment systems of like designs into domains or subnets.

The most common compartmentalization technique used is the corporate perimeter, such as the DMZ which provides controlled separation between an organization's internal systems and the public internet through the use of routers and firewalls.

Compartmentalization is not the same as isolation; rather it is a perimeterization control which uses policies to achieve the following functionality:

- **Access Tollgate**: Beyond just ensuring the identity of the entities allowed access, the ZT tollgate also considers the contextual information surrounding the entity requesting access and may decide to deny access or limit access using a tiered operability implementation.

- **Manage Information Movement**: In a ZT environment, entry does not guarantee capability. All data requests subject to modification or bound for exfiltration needs to pass entitlement checks as well as logged and monitored for future inspection and review.

- **Restrict Lateral Movement**: Compartmentalization affects not only the perimeterized resources, but also movement within those resource. Restricting the Flow Path within the compartment

*Zero-Trust compartmentalization should be applied at the **lowest feasible granularity**.*

It was previously discussed that Zero-Trust defines the asset as the perimeter, but when faced with brownfield systems, decisions must be made on how to begin the ZT journey, what a realistic target state looks like, and the roadmap to get from current state to target state. This will inevitably spark the discussion around grouping of difficult legacy systems into segments or enclaves. An organization's strategy towards their existing mashup of network models should address these major design questions directly and early, as ignoring the issue will increase the architectural headache as time progresses.

Asset Layout & Grouping

The idea of grouping systems and treating them as one organic asset is tempting, as it alleviates the pain of seemingly insurmountable challenges with implementing Zero-Trust in the brownfields. But this exercise must be well thought out and planned. Each grouping discussion must:

- Meet a clear set of criteria for selection,
- Follow a detailed set of guidelines for security and operations,
- All factors surrounding the decision must be documented,
- A migration plan to individual asset-based perimeterization needs to be in place,
- Term limits must be set for periodic re-review and re-approval.

Grouping of systems into a single perimeter should be considered a temporal step in attaining the ZT objectives. At some point, as time progresses, the roles and rules used to protect the grouping will splinter and become unmanageable, so a concerted effort must be taken to monitor and be proactive with grouping.

The most important part of executing a grouping tactic is establishing hardline criteria to determine if grouping should occur, what should be included in the proposed group, and how that group will securely work with other assets. Initially, we envision the following potential areas for grouping, although there may be more reasons to group.

- **Regional Enclaves:** Managing a set of systems specific to regional oversight is an easy way to limit region-specific regulatory requirements from infiltrating the entire organization. This is most prevalent in systems running in China and Russia, where cross-border data exchange is severely limited and monitored.

- **Vendor Segregation:** Delineating vendor-hosted systems allows internal IAM systems to be well protected and allows focused Zero-Trust attention to be paid to vendor-managed access controls. The main issue is the security, monitoring and verification of data exchanged between internal and vendor-hosted systems.

- **Workload Autonomy:** Managing applications in the same functional stack as a single workload can allow simpler management of role-based access control and aggregate like-entitlements. The largest discussion surrounding functional grouping occurs when one or more applications in the stack are ZT-capable, but the decision to include it in a grouping may forego its migration to ZT. This is a balancing act between applying the grouping versus refactoring those non-ZT-ready applications.

- **Legacy Isolation:** For systems that do not have the built-in capability for either fine-grained entitlements or managing data controls, wrapping such systems with an access proxy may be the answer. If such systems are prevalent in a specific department, then it may be a departmental grouping, rather than a single legacy system. However, this does not absolve the need for refactoring the legacy system, as security via proxy is not risk free, so an it must be determined: (1) what the accepted risks are and (2) who will responsible for accepting and managing those risks.

- **Delayed Subnet Migration:** Managing entire subnets as an individual asset may seem like a great phased approach to a ZT target state; but we all know the horrors of interim workarounds becoming permanent solutions. Similar to the legacy discussion above, controls at the subnet level do not have enough fidelity to be considered secure. However, if such an approach is necessary to move the entire organization forward towards a ZT target state; then it should be done with rigorous guidelines around migration, and the pre-commitment of resources to do so. And even with those paper reassurances, subnet grouping may end up being just another corporate appeasement … so beware of such shirt-sighted decisions (and who is endorsing them).

Although grouping can be seen as an abomination of the purist Zero-Trust model, remember that IBM Autonomy model planned for such situations. In such cases, the responsibilities, attributes and operations of compartmentalization are more important than ever.

A Prime Candidate for Grouping

An appropriate area for grouping is the explicit segregation of tertiary enterprise assets such as industrial control system (ICS) components into isolated subsegments from the operating assets of the organization – servers, applications, workstations, etc.

Although the concept may be easy to visualize, the scope of this grouping is not trivial, as there may need to be several subgroupings to restrict lateral movement within the entire segment (and prevent an operational disaster). Below is an example assessment of ICS categorization, and the organization could decide to subsegment by device category, deployment scope, risk classification, or overall impact:

→ Risk: Asset: ↓	Life Safety	Environmental HazMat	Equipment Damage	Material Theft	Data Leakage	Regulatory Legal	Business Operations	Overall Impact
Global (Management Systems)								
Power	Low	-	Low	-	-	-	High	**High**
HVAC	Low	-	Low	-	-	-	High	**High**
NFORMS	Low	-	Low	-	-	-	High	**High**
Data Network	-	-	Low	-	-	-	High	**High**
Telecom	Low	-	-	-	High	Medium	Medium	**Medium**
Video & Surveillance	Low	-	-	High	Medium	Medium	Low	**Low**
Badging & Entry	Low	-	-	High	High	Medium	Low	**Low**
Site (Concentrators & Aggregators)								
Fire Safety	High	Medium	Medium	-	-	-	Low	**Medium**

→ Risk: Asset: ↓	Life Safety	Environmental HazMat	Equipment Damage	Material Theft	Data Leakage	Regulatory Legal	Business Operations	Overall Impact
Power Management	Low	-	Low	-	-	-	High	**High**
HVAC Management	Low	Low	Low	-	-	-	Medium	**Low**
Individual (Devices)								
HVAC	Low	Low	Medium	-	-	-	Medium	-
Power (Distribution, Batteries, Generators)	Medium	Low	Medium	-	-	-	High	**High**
Data Network	-	-	-	-	High	-	Medium	**Medium**
Cameras, CCTV	-	-	-	Medium	-	Low	Low	**Low**
Telecom, VoIP	-	-	-	-	High	Medium	Low	**Low**
Satellite & Broadcast	-	-	Low	-	Medium	-	Low	**Low**

Source: Business Impact by ICS Category, Review of ICS/SCADA Risks[20]

Microsegmentation

Once a layout of asset compartmentalization has been drafted, there are a variety of way to execute a ZT-capable microsegmentation plan. There are many approaches to microsegmentation but between the branding of terms and the architectural nuances, it can be very confusing.

- **Software-Defined Networks (SDN)**

  First conceived in 2011, SDN is an abstraction that “seeks to separate network control functions from network forwarding functions.”[21] SDN looks to orchestrate the configuration of a network infrastructure dynamically from a central control plane. OpenFlow is an open-source reference SDN framework that defines a vendor-neutral standard.[22]

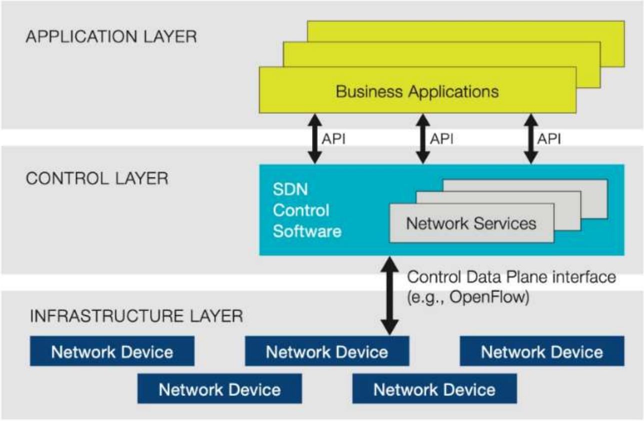


Figure 15: SDxCentral.com

- **Network Function Virtualization (NFV)**

  Cisco differentiates NFV in that it "seeks to abstract network forwarding and other networking functions from the hardware on which it runs."[23] The nuance here is that NFV looks to replace the services available to a network with a programmatic environment that uses commoditized hardware to provide dynamic network capabilities.

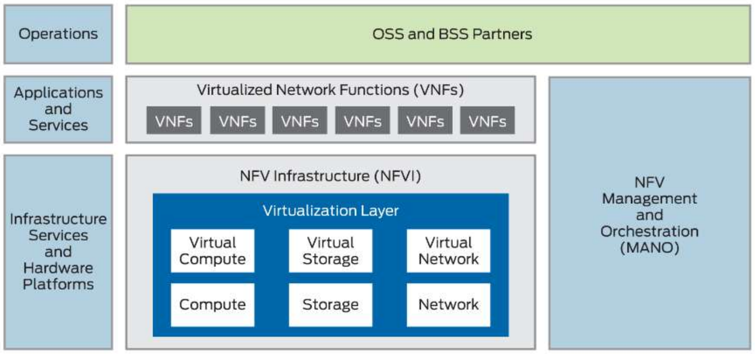

Operations
OSS and BSS Partners
Applications and Services
Virtualized Network Functions (VNFs)
VNFs
VNFs
VNFs
VNFs
VNFs
VNFs
NFV Management and Orchestration (MANO)
Infrastructure Services and Hardware Platforms
NFV Infrastructure (NFVI)
Virtualization Layer
Virtual Compute
Virtual Storage
Virtual Network
Compute
Storage
Network

Figure 16: Juniper Networks

- **Virtual Network Functions (VNF)**

  VNF refers to a set of virtualized low-level operations aggregated to emulate a single network function. The nuance here is that a VNF can be statically defined as virtualized network device, or an NFV solution can dynamically configure a virtual network device with a VNF. To complicate this concept, a VNF can consist of components distributed across multiple virtual machines to achieve its desired functionality.

- **Software-Defined Perimeter (SDP)**

  SDP, also called a "Black Cloud" … evolved from the work done at the Defense Information Systems Agency (DISA) under the Global Information Grid (GIG) Black Core Network initiative around 2007.[24] SDP is a totally virtualized network model that provides just-in-time connectivity based on a dynamic centralized decision engine. SDP is infeasible to introduce into an existing network; it aligns best with a cloud-centric infrastructure.

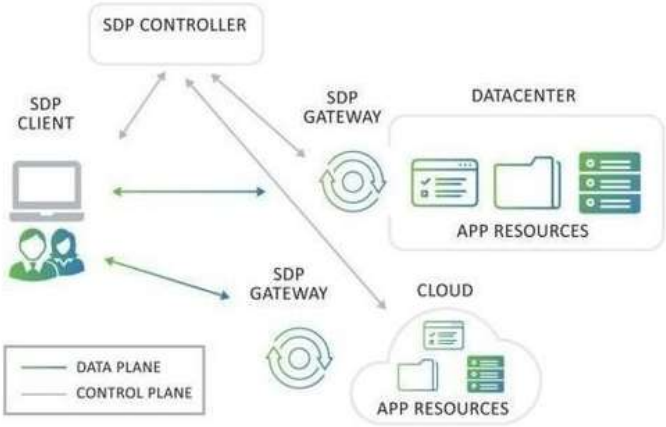


Figure 17: ColocationAmerica.com

SDN and NFV are complementary technologies and as a combined approach they can provide highly granular access to individual networked resources, thus providing functionality similar to that of SDP.

The key concepts to take away from a microsegmentation project are: (1) properly defining what is segmented, and (2) ensuring the decision engine can integrate into a broader ZT control plane implementation.

The Control Plane

For any Zero-Trust incantation, the control plane will be the most frustrating and difficult component to implement. Of course, if you listen to any number of vendors, they'll provide you something they call a control plane, but in reality, it is a proprietary policy administration console specific to the vendor solution.

At a minimum, the ZT control plane needs to:

Address the fundamental mutually assured asset relationships.

- **Actor to Actor**: ex. User to Application, Application to Database, Application to Application
- **Conduit to Conduit**: ex. Endpoint to VPN, Endpoint to Server, Server to Server
- **Actor to Conduit**: ex. User to Endpoint, User to VPN

Note: Many vendors can support subsets of these, and an organization would need to identify where the gaps remain.

Orchestrate multiple layers of networking to manage inter-asset access.

- Ingest data from a wide variety of asset sensors, normalize, analyze, decide and enforce back to any number of asset controllers

- Control routers, gateways, firewalls, load-balancers, proxies, et al.
- Implement or overlay an effective combination of SDP, SDN or NFV.

Provide high availability and resilience.

- Distributed and redundant operations.
- Ephemerality?

Provide defendable operational outcomes.

- Contextualized, but traceable, decision making.
- Common decision logic across instances.
- Unified PDP (policy decision point) and PAP (policy administration point).

Be secure …. really really secure.

- The control plane is the central aggregation point for massive amounts of inference data; therefore, it becomes a high-value target.

- Protect a plethora of confidential sensor data, resource configurations and policies.

One of the most important questions an organization must consider is whether the control plane is treated as "pet" or "cattle" – as that will determine the specific hard protection measures of the implementation.

Unfortunately, in today's marketplace, there is no control plane standard as no vendor specifically supports a ubiquitous control plane implementation.

Organizations need to band together and form a ***coalition*** *to properly define the control plane standards: operational APIs, vendor-agnostic data interchange formats and orchestration frameworks.*

Within the next decade, we surmise there will be two or three major independent control plane implementations, which security vendors will integrate to. But until then, organizations are left to hobble a semi-functional control plane from the vendor space.

Myths, Misconceptions & Assumptions

In the process of defining what Zero-Trust is to your organization, one should also embark on the exercise to define what Zero-Trust should not be. Below are some common myths, misconceptions and assumptions that result in making bad decisions or committing to the wrong path.

Myth: An Organization with ZT can Decommission Firewalls, WAFs and IDS/IPS appliances

For many followers of Forrester's writings and BeyondCorp's reviews of ZT, an organization may be tempted to offset the long-tail cost of ZT through the notion it can remove existing security controls, uprooting years of deep perimeter-based tooling. But the reality is far from that:

- ZT defines the use of dynamic controls executed at the resource level; and distributed & centralized security controls are not mutually exclusive.
- ZT does not inherently fix poor security practices, so if existing controls are weak or policies are obtuse, then a ZT implementation is not going magically make security happen.
- ZT does not eradicate necessity for existing controls such as: security by design in the SDLC, IAM/RBAC definitions, DLP monitoring, et al.

ZT promotes a different paradigm for securing assets, complementing existing controls and uses that information to make more informed context-based decisions.

Misconception: ZT Prevents all Lateral Movement

The goal of ZT-based security is to prevent and minimize the impact of any breach to each asset. This implies the prevention of any unauthorized later-based movement within a network. However, this should be taken as a task, not an inherent right:

- The ability to limit lateral movement is dependent on the construction of the ZT environment.
- Both IAM and microsegmentation increase an asset's barrier-to-entry, but simply increasing the barrier-to-entry is not the same as asset protection.
- Utilizing multiple instances of the same protection mechanism is NOT "defense-in-depth" … i.e. exploits used to compromise one vendor's solution can be re-used against other instances of that vendor's solution.

Preventing lateral movement is an explicit objective that should be part of any ZT planning. The resulting design will most likely include some type of layered security.

Misconception: ZT Ensures Confidentiality, Integrity & Availability (CIA)

It is important to remember that many ZT-capable vendor solutions focus on protecting access to either users, resources or data.

Protecting an asset's accessibility does not imply confidentiality, integrity nor availability.

- Protecting an asset's confidentiality does not imply integrity nor availability.
- Protecting an asset's integrity does not imply confidentiality nor availability.
- Protecting an asset's availability does not imply confidentiality nor integrity.

Distributed Denial of Service (DDoS) attacks can (and will) still occur.

- *Side Note: Quantum Key Distribution (QKD), the exchange of data using quantum cryptography, promises that any encrypted data received between two quantum computers is guaranteed to never have been viewed or tampered, as that would cause the qubits to be discarded, and a "retransmit" attempt to occur. Instead of trying to break quantum-encrypted communication, a hacker could easily attempt to perpetually tap the signal, disrupting the communications indefinitely; hence a denial of service.*

The principles of maintaining information CIA need to be considered as an integral requirement to the ZT design.

Assumption: ZT Protects the Entire Organization

Zero-Trust is a solution to a Risk problem not a Technology problem; therefore, relies heavily on human behavior to comply to security policies and controls.

- Exceptions to ZT policies and rules become pinpoint attack surfaces which degrade the organization's security posture over time.

- ZT can protect an organization's access to the supply chain, but the supply chain itself is an independent organization, and still an attack surface.

Organizational security is only achieved when everyone practices the good security hygiene, regardless of ZT or traditional security measures.

Assumption: ZT Secures the Entire Network

Zero-Trust provides a type of mesh security model to achieve a better security, without long-term needs for refactoring. This, however, does not imply perfect security, and ZT requires all technology assets to be running at optimal security levels.

The mesh security model means zero tolerance for any type of security gap.

- With a distributed mesh security model, there is no central responsibility for security; therefore it is easy for a security gap in a single device to go undetected (although the impact will be limited).

Asset-centric security requires SOC analysts to be omniscient to all asset activity.

- Security operations need to accommodate increased fidelity and volumes of sensor data

Lapses in required security maintenance (Patching, AppSec, et al) will result in breaches.

- *i.e. A small hole may go unnoticed in a partially inflated balloon, but it's almost impossible to put a small hole in a fully inflated balloon without catastrophic results.*

ZT does not obliviate the need for thorough technology hygiene, but rather increases the importance of doing what organizations should have done all along.

The Forgotten Dependencies

Of all the books, presentations and panels on Zero-Trust, most of the focus is on the core components of IAM, microsegmentation and the control plane. Dealing with identities & access, network resources, and the control plane are all first-world issues.

Almost no attention is paid to the downstream second- and third-world affectations on the rest of the organization. These scaffolding dependencies must be pre-planned to ensure other departments aren't left with irreparable harm to their operational responsibilities. I use "scaffolding" as the descriptor because many of these dependencies have indirect consequences to business operations, but also to clients, inter-business transactions and potentially economic reverberations.

Second-World Affectations

Second-world problems are those that are a direct result of changes created by an outside influence. The second-world party may have prior knowledge of the change but did not have any authority over the change.

When planning for a Zero-Trust initiative, it is incumbent on the core team members to understand and address known second-world affectations.

Network Resource Scalability

It’s been well documented by a variety of studies that teams supporting technology are taxed by the complexity of rules. Finding resources that have the skills and experience is becoming more and more scarce. Network teams need to deal with multiple vendors’ firewall nuances, understanding both the language as well as the ordering of rules. It is understandable that each vendor of network devices has their own certification processes to identify professional proficiency.

With Zero-Trust, this volume and noise exponentially increases as there is configuration and log data coming from various levels of ZT resource control layers: (legacy) network perimeter, segmentation layer, hardware/devices, host-OS instances, hypervisors, (virtualized) guest-OS instances, container instances, cloud/virtual orchestration, and finally the ZT control plane components.

Given the complexity of existing rules across proxies, firewalls, routers, load balancers; existing systems have an *unmanageable number of* micro-controller definitions. The organization needs to address this explosion of configuration management and log data generated by all assets.

Cyber Security Operations

Dovetailing off of the scalability of network resources management is the increased complexity and volume of security events being ingested into the organization’s SOC tools. Handling security alerts in a timely manner is only usurped by trying to effectively navigate through the noise. SOCs will experience new challenges in almost every area of operations:

- **Ingestion**: Events will be generated by perimeter devices from traditional security controls, as well as by each ZT-based asset that comprises self-governing security and access controls.
- **Analysis**: Multiple sources of data can also mean that a single rogue event could generate a litany of interrelated events, creating a domino effect which could muddle the analysis as easily as it could enhance it.
- **Enforcement**: A ZT architecture creates levels of control enforcement at the individual asset level, which can result in scalability issues that can affect both operability as well as security.

Zero-Trust necessitates new tactics for SOC operations.

The keys to effectively managing security events is to:

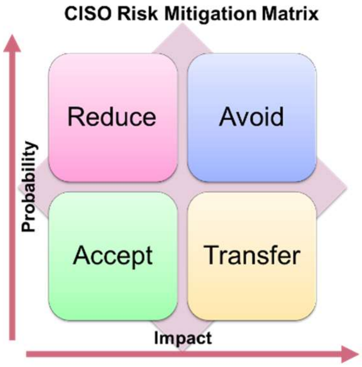
1. Aggregate events centrally through the control plane interface,
2. Use automation to create "threads" of related events, and
3. Employ AI/ML assistance to help assess thread severity.

Determining the success of a Zero-Trust strategy requires that cyber security operations can be effective within the ethos of this new enterprise environment.

Risk Exception Handling

In a traditional enterprise risk management system, there is the concept of risk tolerance. An enterprise uses risk tolerance to determine whether an exception to existing policies requires one or more of:

- **Avoid** the risk by removing the root cause
- **Reduce** the risk to acceptable levels using mitigating controls
- **Transfer** the risk to another owner
- **Accept** the risk **temporarily** with a plan for compliance
- **Accept** the risk **permanently** as a facet of doing business



Zero-Trust significantly restricts the ability to accept risk, dictate the level of risk reduction needed, and, in some cases, remove the ability to transfer the risk.

Zero-Trust is only effective when all the foundational components are operating in concert. Exceptions affecting those components disrupt the protective coverage by allowing holes in the security fabric.

Remediation & Patching

Two specific areas of risk exceptions are the deferment of remediation and patching of systems. These items are called out separately as they are related, present a huge problem, and require a large coordinated effort to manage.

In short, the existing remediation process must change to align with ZT-based migration, patching must be up-to-date, and "Risk Accept" exceptions/observations should no longer be an option for ZT assets.

Desktop Support Services

From the time a ZT initiative starts, there will be two significant impacts to the desktop support services (DSS).

- **Resource Capacity:** When any new control is enabled – whether part of ZT or not – an organization can be guaranteed to have a significant uptick in the calls to helpdesk.

Remote Desktop Accessibility: Part of the playbook for DSS is to first walk the employee through remediation steps; but when self-help does not solve the issue, they initiative a request to remotely connect and take administrative control over the desktop to perform first-hand resolution steps. This role requires ubiquitous elevated privileges, and unless carefully monitored, DSS roles will be a key target for threat actors.

Brownfield Issues

As stated earlier in this document, not all systems are ZT-capable, but an organization must reject any propensity to bundle and manage these non-capable systems as one large ubiquitous segment. This path of least resistance will inevitably cause design and security issues further down the road, as:

- **Communication**: Many ZT-capable upstream and downstream systems now must have access control exceptions to allow communication into a large black box.

- **Data Protection**: This single large black box will have to accept a wide variety of roles and entitlements, so any DLP solution would be rendered useless across this boundary.

- **Eternal Purgatory**: Once abstracted, these legacy systems will all but be forgotten; any migration or complex refactoring may never occur.

As stated in the section “Asset Layout”, groupings are inevitable, so but must be explicit, purposeful and forward-compatible.

Endpoint Access

The mantra of Zero-Trust of “no perimeters” brings with it the paradigm shift from an endpoint accessing the network to endpoint accessing another endpoint. In the purist view, this carries the implication that the term **endpoint access** must be redefined as a mesh or fabric design, as the notion of a VPN-controlled perimeter disappears.

Defining the management of a mesh access control system is infeasible; the realistic approach is to augment traditional VPN and remote access solutions with a system that can proxy access without a choke point.

Zero-Trust is, at its heart, defense-in-depth.

This means that the existing perimeter controls – such as VPN, CASB and VPC – remain in place as a coarse-grain authentication, allowing the more fine-grained access controls to develop over time on the road to the ZT target state.

Endpoint Security

Congruent to the paradigm shift with endpoint access is the increased complexity of reimagining endpoint security. A Zero-Trust target state on endpoints has major ramifications on traditional endpoint controls, moving in general from local static rules to distributed dynamic decisioning.

- **Logging, Monitoring & Alerting**

  Shift from traditional workstation agents and local access monitoring to aggregated behavioral assessment from a variety of direct and indirect sources.

- **Least Privilege**

  Shift from delivery of GPOs and local execution of user policies to remote decisioning (at the control plane), potentially adding latency to every access operation.

- **Data Loss Prevention**

  Shift in securing user-centric activity locally via WAP scripts to contextual protections based on a combination of user actions and data lifecycle.

IoT (lack of) Security

Internet-of-Things (IoT) is a term used to describe any hardware that receives and transmits data for a single intended purpose without the need for a human or a user interface. This includes environmental sensors, medical devices, “smart” appliances, mechanical actuators as well as specific embedded controllers in mobile devices.

Embedded controllers exist in almost every electronic device and are used to assure proper operation. The capabilities for these controllers have expanded to report existing conditions, accept external reconfiguration and execute command requests. Over time, these purpose-built controllers were replaced with more generic micro-processors – programmable logic controllers (PLCs), application-specific integrated circuits (ASICs) and field-programmable gate arrays (FPGAs) – that could be reprogrammed as new functionality emerged.

Not specific to Zero-Trust is the ongoing challenge of securing the organization's cache of IoT devices. Because most of these devices had extremely small footprints, they were limited in supporting functionality not essential to the operation of the device. Absent were access controls and other security measures.

As IoT in everyday devices has grown, so have the opportunities to hijack these devices for a wide variety of nefarious activities. From your smart television watching you and your family, to a massive set of home devices working together executing a DDoS attack, IoT provides an unmitigated channel for malicious threat actors to operate from.

Introducing Zero-Trust into the infrastructure must account for the inability to protect IoT.

What a ZT architecture can provide is threefold: validate/filter any input requests to the device, verify/sanitize all output from the device, and limit lateral movement from any particular IoT connected to a corporate resource.

Internet Access

Once there is no longer a distinction between the DMZ/LAN and corporate resources hosted on the internet, what constitutes protecting users from unapproved sites quickly dissolves. The usefulness of traditional firewall rules for blocking outbound access to bad IP addresses and suspicious hosts is nullified; hence, the balance moves to critical reliance on website blocking mechanism via services such as Bluecoat web proxy categorization for public internet access safety.

For scalability, these services index sites by category and allow enterprises to block full categories; but at the expense of managing explicit exceptions. For example, if an organization needs to allow access to their Microsoft O365 SharePoint repository, it could traditionally block the category of "file sharing" but allow the exception to "acme.sharepoint.com" at the firewall. With ZT, these decisions are relegated to the control plane, which must align the user with their active role with the specific data to be transferred with the target destination with the geolocation of the endpoint with various other factors.

This might be feasible for an organization with ~1000 employees, but quickly expands out of control as the workforce reaches 25K or more.

Third Party Systems

Even if an organization has implemented SAML (or exposed LDAP) to their supplier-hosted systems, these third-party systems may not migrate cleanly to a ZT-architecture.

Externally hosted and managed systems are a blind spot for any Zero-Trust implementation.

The problems here are threefold: (1) there is no control over what a vendor does with its information or how it is protected at the asset level, (2) there is no ability for the organization to inspect or ensure compliance to their internal ZT requirements, and (3) an organization cannot restrict lateral movement outside its own hosted/managed assets.

These problems become more uncontrollable as many third-party vendors use outsourced applications, platforms and services; transferring the risks to fourth and fifth parties, thus further distancing the organization from its ability to control the risks.

Vendor Access to Corporate Assets

Compounding the vendor relationship is the ability to control external access to corporate assets. This frequently occurs when on-premise solutions and hardware need to be maintained by the manufacturer.

Securing access becomes more challenging when the provider outsources its 1st- and 2nd- tier support, making identity verification more obscure.

Third-World Affectations

Third-world problems are those that are an indirect result of changes trickled down from an outside influence. The third-world party has no prior knowledge of the outside conditions as they are too far removed from the original change decision.

When planning for a Zero-Trust initiative, the following third-world issues can be mitigated by involving key team members from other areas of the organization.

GRC, Policy & Control Management

Although Zero-Trust relies heavily on uplifting the technology and security infrastructure of an organization, it really is a business risk effort more than a technology risk effort.

Planning for Zero-Trust must include a top-to-bottom reexamination of both operational as well as security policies.

When business risks are reevaluated, it should lead to a parallel reevaluation of the highest levels of risk management: corporate policies. Unfortunately, in many cases, fundamental technology changes get implemented without regard for any misalignment to current policies.

For example, implementing FIDO or passwordless authentication may ironically violate password policies, and possibly MFA controls as it could be considered a single factor.

There must be a period of time where an organization needs to employ multiple policies, supporting both legacy activities as well as zero-trust activities. The key here is to augment each policy, existing and new, with conditions surrounding their enforcement.

Data Loss Prevention

A mindful Zero-Trust architect should examine how their organization's DLP operates and identify any potential leakage gaps that need to be addressed with additional coverage. DLP tools tend to have three major flavors: (1) perimeter triggers, (2) user triggers, and (3) data triggers.

- **Perimeter Border Crossings**

  The traditional perimeter-based control which works reasonably well within a controlled environment, but fails miserably as organizations blend cloud-native, cloud-assisted and legacy environments thereby erasing any clean delineation of a perimeter.

- **User Behavior Activities**

  Where perimeter-based security fails, the next feasible attempt to prevent data loss is to monitor the actors that cause data leakages. Many of these solutions require endpoint agents, which implies all users will be using managed devices. But workarounds are prevalent with the growth of SaaS providers running applications independently in the cloud. Many zero-trust vendors that purport the "user is the perimeter" have doubled down on this type of tactic.

- **Data Lifecycle Tracking**

  Idealists of the data lifecycle management world would like to see every data element have metatags defining the confidentiality, origin, time-sensitivity, and other contextual attributes. With all this information, systems could better track and prevent entitlement violations. This practice would require an enormous datastore for metadata, which poses problems of scalability, synchronicity and usability. Realists in this area submit that only critical data needs to be managed, which becomes a more feasible problem to solve. Tracking data movement also has its workarounds, as logical boundaries of monitoring would still exist. As above, zero-trust vendors that purport "data is the perimeter" have doubled down on this type of tactic.

A robust DLP solution requires all three tactics, and even better, if these tactics could communicate with each other to build a larger picture of the data movement. As with other areas, orchestration becomes a critical requirement for the zero-trust variety of this control.

Automation

The advent of our automated world has brought with it leaps in progress as well as new types and levels of threats. Automation needs to be examined under five different lenses: process optimization, task commoditization, data scraping, automation platforms and attack swarming.[25]

- **Process Optimization (RPA)**

  The accelerated use of automation in business has both significant benefits and challenges to organizations in general – not just the Zero-Trust initiative. Unattended server-hosted automations, aka robotic process automation (RPA), challenge some of the basic Zero-Trust principles:

  - Violate policies and/or regulations pertaining to credentials and elevated privileges
  - Lack of scalability, fallback, resiliency or continuity plans
  - Implemented without awareness for downstream impacts
  - Data combinations move from benign to confidential

- **Task Commoditization (RDA)**

  Closely related to RPA is the use of localized automation at the individual task level. Whereas a formal RPA project follows specific rules of the software development lifecycle (SDLC), robotic desktop automation (RDA) – individual user-attended automations at the desktop – flies well under the radar for policy inspection and control mitigation. RDA – also fondly referred to as "programs by non-programmers" – brings its own unique set of issues, in addition to the RPA issues above:

  - Replicates bad habits/workarounds/shortcuts
  - No thought to discourse/error handling
  - Introduction of unmanaged/undocumented business logic
  - Inadvertently violate policies and/or regulations
  - Lack of maintenance and patching resources

- **Data Scraping Legalities**

  In the early 1990's, automation was originally used by organizations to gather unstructured data from websites. In today's age of REST APIs and other web services, it seems that data scraping is a crude use case. However, it brought to light semi-religious arguments surrounding the legalities of electronically capturing one organization's generated data and its ingestion and usage by another organization.[26] There are several safeguards that automation needs to abide by, which would include any automation used by ZT that ingests externally owned data:

- Content being scraped is not copyright protected
- The act of scraping does not burden the services of the site being scraped
- Do not violate the Terms of Use of the site being scraped
- The scraper does not gather sensitive user information
- The scraped content adheres to "fair use standards"

- **Automation Platforms**

A frequently overlooked concern with automation is the policy exceptions created by the deployment needs of the various automation platforms. This can range from use of "service-IDs" performing the same tasks as a human without the accountability measures, to the elasticity methods which may not be ZT-compatible.

- **Attack Swarming**

Externally, automation has become a mainstay for threat actors. This increases the need for asset-level protection paradigms such as Zero-Trust. We already see several trends coming to light with automated attacks:

- Click Farms have increased exponentially
- Automation has commoditized TTPs, sophisticated attacks are more prevalent
- Swarming of containerized command & control (C&C) servers makes tracking and attribution almost impossible

Not all automation is a stressor to Zero-Trust. In fact, many of the PIP and PEP operations require automation to operate consistently and at scale. AI/ML engines use automation in much the same way: bulk data gathering from a large breadth of sources, and decision enforcement to a large variety of targets.

Embrace automation in a ZT environment with the same level of due diligence as any other large RPA project.

When an automation fails, it will fail fast and wide; having plans for continuity, remediation, fallback and recovery are imperative. When your organization's security depends on automation, small problems can become exacerbated, so extra care must be given to design patterns using automation.

AI / ML Projects

Much of the ZT control plane's PDP will use machine learning for sensor analysis and artificial intelligence for providing viable alternatives in the decision engine.

The area of AI/ML that is of concern is the black box computing done at the operations level – those business processes that ingest, manipulate and output potentially sensitive data without the ability to inject fine-grained Zero-Trust entitlement checks.

Most FinServ organizations are already dealing with a similar issue due to the regulatory requirement for financial instruments and operations that "*each decision must be understood well enough that it can be explained to a regular person*."[27]

Having an inventory of AI/ML instances, documenting their algorithms (if known), and tracking the input and output data elements can help improve the enforcement of ZT policies and controls.

DevOps & DevSecOps

DevOps (and DevSecOps) is the notion that a developer will be the best person to debug and remediate any operational (and security) issues with the application they've help build. This implies the developer will have access to production systems and data.

The popularity of DevOps and DevSecOps roles in an organization is well-deserved: it has proven to be an effective tactic in delivering quicker support response and resulting in better resolution quality, all while reducing costs.

But where Zero-Trust promotes the notion of "segregation of duties" combined with "least privilege," the precept of DevOps and DevSecOps flies directly against those two security tenets.

An organization cannot do both Zero-Trust and DevOps-Trust without significant challenges.

One possibility is to allow endpoints to give developers a "context-switch" to change roles. This will be complicated as LDAP supports multiple role assignments but cannot make runtime distinctions for entitlements.

Consider this alternative: Providing a single user with multiple identities solves the user verification and entitlement issue but violates the single-user/single-identity policy … and still allows for data cross-contamination on the local endpoint. Thus, to prevent data cross-contamination, a developer must have separate endpoints to access separate roles using separate identities. The question remains how to handle developers that support more than one application, potentially creating a "toxic combination" of elevated access.

There is no straightforward solution to architecting a Zero-Trust infrastructure that securely empowers DevOps-Trust … and simply "doing nothing" should not be a valid option.

SDLC, Build Environments, CI / CD

Regardless of the software development lifecycle (SDLC) employed by an organization – sometimes numerous SDLCs – a ZT architect must investigate how the SDLC is applied and what specific access touchpoints each developer has. In addition, every methodology from Waterfall to Agile/Lean needs to be reviewed under the ZT microscope for areas that might allow unmitigated lateral movement, unauthorized access to data, or privileges elevated beyond what is needed by the developer role.

Policies around any SDLC need to support:

- **Virtualization, Containerization & Micro-segmentation Controls**

  With current virtualization technologies, there is the need for developers using local virtual environments for development to require administrative access to their endpoints. This opens the door for unapproved applications to be installed, malware escaping the hypervisor/sandbox, and abuse of privileges by the insider threat.

- **Required Use of Test Data**

  There is prevalent use of production data for test environments, which manifests two major threats:

  - Test environments are typically less secure than production environments.
    - This can lead to data loss of improperly protected sensitive data.
    - DLP investigations may find it difficult to pinpoint the source of leakage, as they may not realize the data was sourced from a test environment.
    - Determining scope of a breach would also prove problematic as monitoring of test environments is absent.

  - Developers access to sensitive production data may go unabated, as they are not scrutinized with the same level of fidelity as the authorized handlers of that data.
    - FinTech developers using production data may be in violation of not only privacy regulations, but also SEC regulations on: (1) pre-trade compliance registration and (2) regulated user monitoring.

- **Segregation of Roles from Employee to Developer**

  Joining an organization as a developer comes with the tools needed to perform one's duties, from an integrated development environment (IDE), local virtualization platforms, secure remote access to special development labs, and many other 'toys" that are not part of the standard endpoint. These tools coexist on the same endpoint as the corporate utilities – email, wiki, file shares, internet access, etc. The implied issue is that there is no segregation of duties when the developer is performing "developer" duties versus "corporate" duties. Thus, it is possible for the developer to take source code, sensitive data, or encryption keys and exfiltrate them using the corporate tools.

Overlaying Zero-Trust requirements onto this combined role makes for a clear security gap. This issue should be taken as seriously as the DevOps issue above but is usually overlooked because it has been the business-as-usual (BAU) protocol for far too many years.

Human Resources

The Human Resources Team is probably the last team you'd expect to engage with a Zero-Trust infrastructure effort, but their involvement with onboarding and offboarding procedures is essential to ensuring all identities are synchronized.

Beyond the first-world issue of proper role/entitlement assignments, there are two other aspects where the HR department can be of assistance: orphaned identities/assets and behavioral risk ratings.

- **Role/Entitlement Assignments**

  HR plays a critical role in assuring the consistency and performing validation of an Active Directory (or other LDAP) identity repository. RBAC can be very complicated, and many organizations' AD definitions are overrun with exceptions and one-off entitlements. HR should be involved in any entitlement exception process and raise flags where it may violate a Zero-Trust principle.

- **Orphaned Identities/Assets**

  The vulnerabilities posed by orphaned identities has already been discussed, and HR has a critical responsibility to ensure that:

  - Offboarding of individuals is comprehensive to include central identity repositories, third-party access authorizations, as well as internal applications using embedded or isolated access credentialing.
  - All service-IDs, applications and supplier relationships owned by the offboarded associate are properly re-assigned to workforce members with *the same role and entitlement level*.

- **Behavioral Risk Ratings**

  Insider threat management is part and parcel of any security framework, and Zero-Trust is no different. Traditionally, ITM and internal behavioral risk rating systems are owned and managed by the security operations team; but HR needs to be a key player in the rules and rating methodologies used in these risk rating systems, as they may have additional information to supply to the algorithm that can greatly enhance the robustness of the rating.

HR needs to be involved, not only in the onboarding of workforce, but in the proper offboarding and insider threat management routines.

Roadmap to Zero-Trust

Defining the ZT target state is a feat in and of itself but achieving that target state can take years. We surmise that as technology progresses, the target state will be perpetually moving forward; so, it behooves the prudent organization to realize that Zero-Trust will be a journey rather than a destination.

The Zero-Trust journey, contrary to intuition, does not mean limitless funding and rework; rather, it means an infrastructure designed to be flexible, dynamic, progressive as well as cost effective. Where traditional security has an ever-increasing cost-to-output ratio, a zero-trust initiative will have a large initial cost followed by a year-over-year (YoY) cost basis that decreases over time.[28]

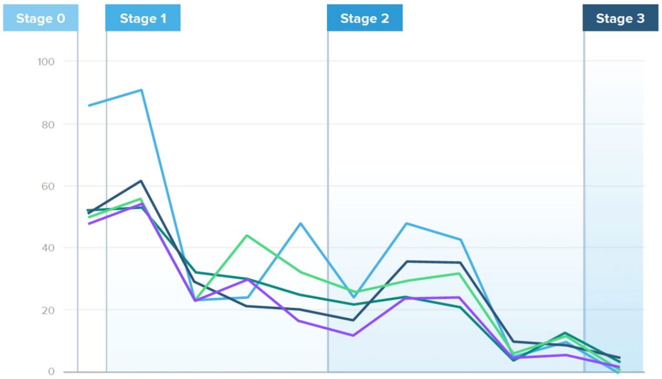


Figure 18: The State of Zero Trust Security in Global Organizations (Okta)

There is a litany of viewpoints from vendors and experts on the proper steps to achieve Zero-Trust nirvana, but the most effective plan is the one that the organization defines itself based on research into the various methodologies.

Defining the Zero-Trust roadmap requires determining the right priorities at the right time.

It is in this vane that we propose our own tactics for building your own roadmap.

Pre-Planning Steps

The obvious prework to any large infrastructure project is a set of clear business justifications but project buy-in needs to span horizontally across business divisions and vertically from executive commitment down to technology support teams.

The sections below are not promoted in any order of preference. As organizations differ in culture, one organization may require executive politicking first, whereas another may require technology preparation before any other activity.

Concerns from Executive Leadership

The section "Why " addressed some of the business justifications for taking on such an arduous project, with the most compelling justification of the continuous drop of ROSI (return on security investment) towards zero. Even with iron-clad preparation, executive leadership will always ask the following probing questions:

Do We Need It?

This may be the toughest question to answer as executives will reference previous quarterly security scorecard reports for comparison. Your response will always present a juxtaposition:

- If the existing security posture is accurate, as reported every quarter, what warrants such a significant multi-year spend?

- If the existing security infrastructure is deficient, then previous security reporting might be seen as flawed, and executive confidence in your ability to properly assess security is nullified.

The best approach to this question is to focus on two key issues:

- **Reducing/Avoiding Technical Debt**

  Technical debt is a well-known organizational issue, and always the topic of technology budgets. By aligning the Zero-Trust initiative as supporting the reduction of technical debt, executives are given an opportunity to amortize the costs of technology uplift requests between multiple budgets – CTO, CISO and business units all bear a portion of the resource costs to uplift applications and devices.

- **Long-Term Impact Compared to Doing Nothing**

  This is where we reiterate the minimizing ROSI statistics, but also compare it to the strategy of not taking any action. This can lead to conversation from the increased frequency of potential breaches (as attacks become more complex) to the lagging progress with competitors in the industry that have competitive advantage based on embracing emerging technologies.

Can We Support It?

When proposing such a large effort, resource planning will always be top of mind. Many times, the message from the top will be to “do more with less.” Instead of focusing on WHAT resources are needed, focus on HOW resources are needed:

- **No Standard Migration Path**

  It is imperative that the executive leadership understands that “Zero-Trust” is a framework, not a material checklist. There is no standard for implementation much less a roadmap for migration that can be accurately quantified.

  However, also come prepared to demonstrate a clear understanding of how your organization defines their ZT target state. Present a well-planned series of milestones (discussed later in this document) that show progress/risks at each stage and the exit/fallback plans if success criteria are not reached, an immovable obstacle prevents progress, or funding dissipates.

- **Shift in Technology, Operations & Culture**

  Reiterate that Zero-Trust is not a security project; rather a holistic strategy for reducing risk. Such a bold initiative requires a shift of focus across the technology footprint, business operations and organizational culture.

  Be honest, though, that there will be a minimum required technology and human resource needs that is unavoidable.

Finish this conversation that “doing more with less” is most effective when everyone assimilates and contributes to the new culture; and this is where the Zero-Trust initiative needs messaging from the top to be successful.

Can We Bear It?

Between the ideation, design and execution of this initiative, there will be advances at both the conceptual level as well as the supporting technologies. Beyond the fiduciary capability to support a ZT initiative, how will quarterly progress reports reflect success or failure with such a fluid paradigm? Educate the senior executive community that the ZT strategies will be constant, but tactics and target state may adapt over time.

- **Flexibility for Emerging Technologies**

Suggest that a test environment to pilot designs and technologies is necessary to ensure an implementation will have the least friction when moving into a production environment.

- **Iterative Development and Growth**

Mapping out the rate of vendor solutions in the ZT marketplace should launch a discussion surrounding the need for patience with iterations.

The key here is to push the idea of "patience" with regard to the immaturity of the marketplace. This, however, should not offset the need to start architecting today for a ZT target state.

Building (and Tempering) Expectations

Creating a team that crosses both organizational and functional boundaries is just the first step to defining a comprehensive ZT target state. As the team brings in its newest members, there may be varying levels of understanding on Zero-Trust – much of it comes from vendor papers – and needs to be dispelled or redirected to the organization's needs:

- **Replicating "BeyondCorp" is an Unrealistic Target**
  - Envision a Hybrid Security Model (not an "Either/Or" Model)
  - Build for Cloud-Native/Container, but allow for Legacy
  - It is a Holistic transformation, not a Wholesale change

- **Normalize Endpoint Access**
  - [mis-]Trust your external and internal users with the same level of due diligence
  - "On-Network" and "Remote Access" merge to become "Unified Access"
  - Reduce user access complexity by proper implementation of MFA+SSO

- **Security-By-Design is Table Stakes for Application Development**
  - Continuous verification (AuthN) and entitlement (AuthZ) checks at key logic checkpoints
  - Implement mutual Application-to-Application verification
  - Create self-aware and/or self-securing applications

- **Find Your Balance Between Over-Simplification vs Over-Engineering**
  - Brainstorm freely to determine all possibilities
  - Use a defendable approach in making every design decision
  - Document all decisions, conditions, assumptions and discarded options
  - Don't be afraid to revisit alternatives when conditions or assumptions have changed
  - Define the criteria/threshold to move forward in the design process

- **Goal Alignment at All Levels**
    - Involve all Business Unit Leaders in roadmap discussions and decisions
    - Provide Education on ZT and enterprise vision & objectives to everyone
    - Messaging spans from Application Developers to Senior Managers

A recent survey found that almost 1/3 of security practitioners were not confident that a Zero-Trust initiative could be successfully implemented.[29]

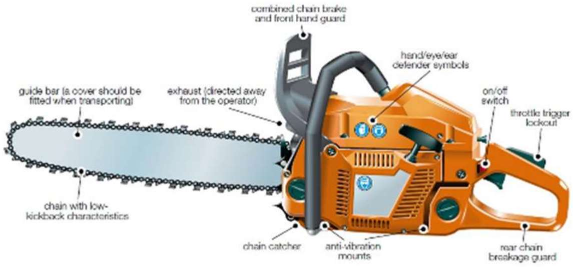
Zero-Trust relies on great leadership as well as great design.

Team support is just as important as the executive support, and a lack of cohesion can undermine any project. The ZT team leader must focus on people as much as they do the design process.

Determining Good System Design: The Chainsaw Approach

The "Chainsaw Approach to Design" was coined by J.C. Checco back in 2010 to simplify the technology evaluation process.

In this scenario, imagine a consumer "Bob" wants to buy a chainsaw. Bob knows he needs a chainsaw because he has researched and found that a chainsaw would be the tool to satisfy his need.


combined chain brake and front hand guard
hand/eye/ear defender symbols
guide bar (a cover should be fitted when transporting)
exhaust (directed away from the operator)
on/off switch
throttle trigger lockout
chain with low-kickback characteristics
chain catcher
anti-vibration mounts
rear chain breakage guard

So, Bob goes to the local hardware store and tells the owner "Alice" he needs to look at chainsaws. Alice and Bob discuss several factors that best determine both form and function, and Alice relays her experience with the brands and models that are the most reliable and effective. Alice also speaks to safety features explaining chainsaw mishaps and how they could have been avoided. Alice helps Bob decide on the brand and model of chainsaw that is designed well, performs above average and has all the recommended safety features. Before Alice will allow Bob to purchase this particular chainsaw, she asks if he actually knows how to operate the chainsaw – and perhaps even allows Bob to demonstrate his competence in using the tool on a piece of lumber in the back lot.

What Bob never told Alice – and Alice never thought to ask – was that the chainsaw was going to be used to cut his annoying neighbor's car in half. If Alice had known this, she would have not only refused to sell Bob a chainsaw, but perhaps recommend a better alternative to rectifying the root cause of the issue between Bob and Chuck (his annoying neighbor).

Although this scenario sounds completely and utterly absurd, it happens quite often – and unintentionally – with various technologies.

The chainsaw approach allows us to ask some very concise clarifying questions about any system design with 6 simple characteristics: **purpose, function, safety, construction, operation** and **usage**.

1. What is the System's **Purpose**?

This is a fairly obvious question but sets the groundwork for subsequent inquiries. Applying Occam's razor, we look for is the simplest and most direct answer.

In this case, the chainsaw is designed to cut wood. But, if you were asking the vendor for a "mouse trap" without providing any context on the purpose, you may end up with a board game.

In the cybersecurity world, a vendor peddling a CASB product may not actually perform all the expected functions of your definition of CASB.

2. Is the System Designed to **Function**?

What is a seemingly innocuous question actually implies a more serious inspection into the design of the tool. In other words, does the design fit the intended purpose?

All chainsaws on the market must be designed to pass ILO manufacturing standards[30] that mandate specifications such as the direction of cutting teeth on the chain.

But software manufacturers aren't required to comply to that same level of rigor, as there is no tangible way to validate the expected functionality without a PoC or pilot. Functional deficits in software are so common that we use monikers such as "snake oil" and "vaporware" to describe them. Many times, the vendor can explain in detail all the additional features of a solution without disclosing that the solution cannot satisfy even the minimum requirements of the main function.

3. Is the System Designed to Support **Safety**?

This question determines if the product is safe to use as designed. Protection measures need to be clearly spelled out and acknowledged as designed in or bolted on.

For the chainsaw, we would want to know that it is designed with safety features such as a front hand guard, a chain brake, dead man’s switch and kickback protection. Any specific items bolted on as an afterthought could severely affect the operation, performance and even overall safety of the tool.

For any security tool, we would want to ensure that the vendor discloses any and all prerequisites, dependencies and side effects. For example, It is counter-productive when a DLP solution requires that an organization pre-classify all its data elements, as that would be an insurmountable task in itself; but the customer-owned requirement exists to because the solution: (1) may not have the tooling to dynamically or accurately identify sensitive data, and (2) allows for plausible deniability when the solution only shows limited success in preventing sensitive data loss.

4. Is the System **Built** as Designed?

All new things have that unique smell, that taste, that brightness. This question delves below the shiny surface to identify areas that may cause us concern later.

For the chainsaw, we want to ensure that it indeed has the proper compression ratio on the motor, the proper tension on the chain, that the chain was installed in the proper direction, that the shut off button exists (and works).

Front End User Experience

Back End Application Logic and Source Code

For our security tools, we need to perform that same inspection. Beyond the streamlined UI and cool animated dialogs: Are there issues with operations at scale? Can the log data format be imported into a Splunk or a SIEM? Is the solution upgradable-in-place, or do we need to reconfigure with every major release? Even questions such as “what programming languages and frameworks were used” can predetermine potential future issues.

5. Is the System Being **Operated** Properly?

After the product has been thoroughly examined, and found to meet the needs of the buyer, we must focus our attention on the buyer themselves. Can the end user properly and safely operate the tool so as not to harm themselves or others?

Unfortunately, it is not the responsibility of the seller to verify if the consumer knows how to use a chainsaw. This explains why there are approximately 36000 chainsaw injuries per year in the U.S.[31]

From a cybersecurity perspective, properly installing, configuring and deploying a tool is table stakes. Misconfiguration or misinterpretation of results can lead to adverse effects in both the organization's operation and reputation.

6. Is the System Utilized for its **Intended Purpose**?

Although we verified that the tool operates for the purpose it was designed for, this does not ensure that it will be utilized for that same purpose.

Again, not the seller's responsibility, but if the consumer was purchasing a chainsaw to cut a car rather than a tree, we can surmise they are not going to be successful.

In the world of technology, such misappropriations happen more frequently than we care to admit. For example, proximity badges are highly recommended as a secondary means of login authorization and for ensuring unattended logoff.

In one case, a hospital equipped all personnel with proximity badges and reversed the usage to allow automatic login, as "*doctors didn't like always having to type in a password*." This bastardized use of proximity access control for medical devices was troublesome not only because it gave a false sense of security, but those sensitive devices were unlocked every time a doctor passed by. Yet, the hospital was considered fully compliant by HIPAA auditors.[32]

This comes down to having well-prepared use cases that exemplify the requirements of the design. In some organizations, use cases are developed from pre-defined requirements, and sometimes requirements are derived from use cases. Each organization has their preferred method of developing use cases; but the salient point here is to ensure all requirements are reflected in at least one use-case.

Maximizing Existing Security Investments

Before embarking on the Zero-Trust journey, one needs to understand the organization's **current state**: its environment, architecture and tooling. From there, one needs to map how existing components serve each purpose in a ZT target state.

If one looks at their existing flow path for a typical request/response operation, they would see individual security components such as: user access (LDAP), user verification (FIDO/OTP), endpoint verification (MDM/WDM), corporate access (VPN), secure end-to-end communication (TLSv1.3), application verification (code signing), data assurance (DLP/ITM), and other measures.

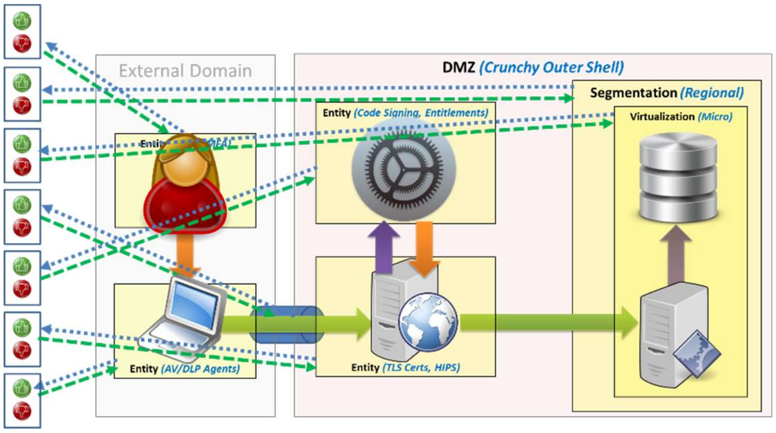


Figure 19: Introducing Zero Trust into an Enterprise Infrastructure, Checco (2018)

At its most fundamental level, a control plane simply aggregates the operations of all those controls through a single portal. This is more easily visualized than implemented as each security tool exercises vendor lock-in: providing its own administration console and proprietary data formats.

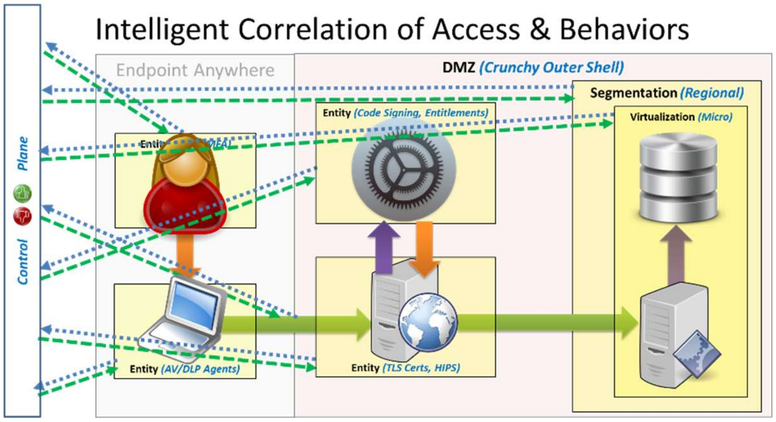


Figure 20: Introducing Zero Trust into an Enterprise Infrastructure, Checco (2018)

Security function mapping may seem like an insurmountable task, but it can be somewhat simplified by employing a cheat sheet of known security measures. The security architecture snippet below[33] by Adrian Grigorof is concise, clear cut and well worth obtaining the full version.

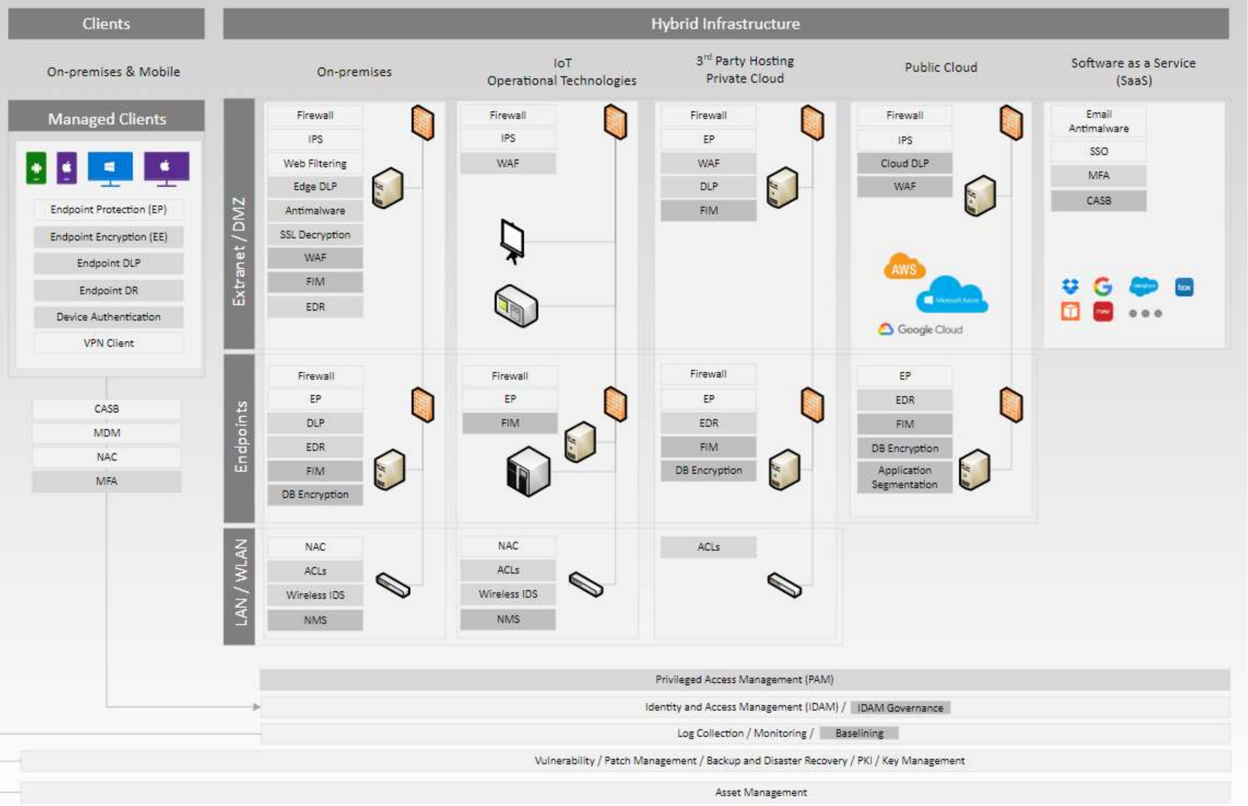


Figure 21: https://www.managedsentinel.com/downloads/one_page_security_architecture_v2.svg

Once there is a mapping of existing capabilities, there needs to be discussion around how far existing tools can be repurposed as integral parts of the control plane – PIP (policy information points) and PEP (policy enforcement points) – rather than just sensors and tollgates:

- **Overlay**: Where do existing functions align to target state requirements?
  - Does a tool capability map to a ZT target state requirement?
  - Do changes in tool configuration affect target state mapping?
  - Is a tool capability fragmented across one or more partial requirements?

- **Requirements**: How do those existing capabilities meet the requirements?
  - Does the tool have unused functionality that can support ZT requirements?
  - Can the tool scale as needed by ZT?

- **Modularity**: Can the tool functionality be used in a ZT-based manner?
  - PIP: Can the tool support API or export of sensor data into a control plane?
  - PAP: Can the tool configuration be remotely managed by the control plane?
  - PEP: Can the tool enforce externally generated decision policies from the control plane?

The magic of a true Zero-Trust control plane is that it not only collects data from multiple security control sources but coalesces that data for more contextual decision making.

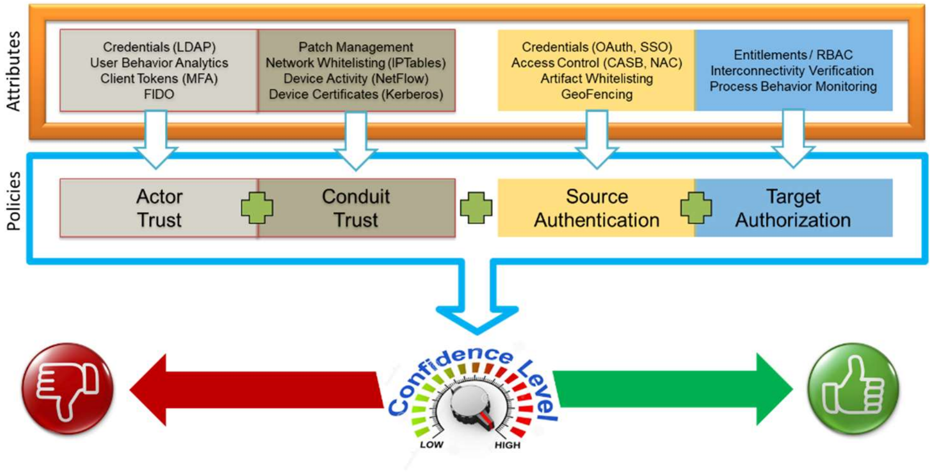


Figure 22: Introducing Zero Trust into an Enterprise Infrastructure, Checco (2018)

With a contextual decision-making process, there is an inherent change in complexity: from an absolute pass/fail result to confidence-based output that must be assessed against a dynamically generated threshold.

Migrating to Zero

This fluid security process cannot be achieved solely with existing security tools, and in most cases, will require concessions with legacy policies, tools and systems. As previously stated, concessions and exceptions do not bode well within the Zero-Trust paradigm; but such abominations are unavoidable, so a phased approach is needed to migrate away from non-ZT-capable systems and tools.

An organization must move towards the Zero-Trust target state without compromising existing security in the interim.

Policies, Standards, Controls & Procedures

Part of designing a Zero-Trust target state will be the re-imagining of security from scratch, which encompasses the top-down security policies from the governance, risk and compliance (GRC) team.

First and foremost, there needs to be a clear congruence amongst all parties on the role and differences between policy, standard, control and procedure.[34]

	Purpose	**Change Frequency**	**Responsibility**
Policy	Objective / Intent	Rarely	Organization
Standard	Quantifiable Requirements	Infrequently	Business Unit
Control	Prescriptive Compliance & Mitigations	Frequently	Stakeholder
Procedure	Detailed Steps, Actions & Responses	Living Document	Affected Individual
Guidance	Additional Information / Recommendations	Living Document	-

It greatly benefits and organization to first assess and refactor their existing policies, standards and control using the above delineation prior to embarking on a Zero-Trust project. This alone will boost the security posture of the organization.

As it is tempting to use the GRC team to generate the ZT-centric policies, they will be biased to existing policies to see clearly what is required for a ZT target state. Conversely, bringing in an independent third-party consulting firm to provide cookie-cutter ZT policies will be just as detrimental to the success of the program.

The balanced approach would be to engage a team of experienced security and technology personnel from across the organization to identify and detail how an ideal ZT target state would work for that organization and build policies around that customized target state.

The ZT-policies would then be defined into further standards, guidance and controls – and then overlaid against the existing control mapping to find where existing controls are aligned, complementary or conflicting.

Once there is a clear understanding where the organizations sites today and where it needs to be, a progression of migration must take place – i.e. at what point does one allow for (and eventually remove) conflicting standards?

Refactor, Rebuild or Buy?

For non-ZT-capable systems, explicit mid- and long-term planning must take place with those application owners. Discussions should cover the cost in time, budget and resources needed to either refactor the system for ZT capabilities, rebuild the system on a ZT-capable framework, or replace the system with a ZT-capable third-party SaaS – and options are not limited to those three suggestions. This also may mean a fundamental shift from an on-premise to cloud deployment which has additional challenges and requirements.

Engaging these application owners also gives them a level of participation entitlement in the ZT roadmap, allowing them to work **with** you – making the change less of an unfunded mandate and more of a cooperative progression.

If your ZT program truly has executive commitment – from a financial / budgeting perspective – there will be an appetite for application owners to collaborate, because funding is the great motivator.

Supply Chain Management

Coordinating your organization's ZT initiative with your supply chain is yet another uncontrollable challenge that must be addressed. The supply chain should be managed in three distinct baskets, from least to most challenging:

- **Support Systems**

  Support system suppliers are those vendors that are: practicing ZT-like measures, immune to the changes that the ZT effort imposes, easily adaptable to the ZT changes, or can be managed independently in the ZT migration. Such systems include facilities management, contractors not needing system access, office supplies and wellness programs.

- **Technology/Security Vendors**

  Although there is great benefit to using (or repurposing) as much of the existing security and technology infrastructure as possible, the ZT planning team needs to be prepared for the feeding frenzy by those same technology vendors once they hear the trigger phrase "Zero-Trust." These vendors will spin or even fabricate reasons why their technologies are fully ZT-compliant, and why you should expand their footprint into your organization. But your planning team should be prudently skeptical of such claims. Many of your existing technologies will satisfy a subset of ZT-based concepts, and that's what we recommend you take advantage of.

- **Operational/Transactional Suppliers**

  The operational and transactional vendors are what most people envision with the term "supply chain." These are the external entities or systems that are directly involved in upstream and downstream business operations; they have a direct effect on the organization's bottom line. These suppliers are often identified in the organization's BC/DR plans as critical partners and have gone through extensive 3rd-party vetting. Because these suppliers have a direct impact on the business, it will be challenging to move them towards a target state they themselves have not defined.

The best course of action is to disseminate your organization's specific ZT implementation requirements to all the vendors and obtain specific mappings where existing vendors satisfy particular requirements, and where new vendors need to be engaged.

*Solutions should be rated on the nuance of **how** they satisfy your organization's interpretation of the Zero-Trust requirements.*

We believe the key to success here is to engage your supply chain vendors early and often. This will give your suppliers the foresight to progress forward, or your organization the lead time to find vendors who are a better fit into the Zero-Trust mindset.

Execution Plan

"Measure twice, cut once" is a prudent plan when dealing with items that are irreversible. But that cliché acts as an anchor to Zero-Trust, as over-engineering can be a real issue. In a ZT world the pace of changes in threats, best practices, and the marketplace is even faster than internet time; so detailed plans for specific technology solutions will execute efficiently, but perhaps not effectively.

NIST's SP800-207 draft on Zero-Trust is purposely **not prescriptive** "as an enterprise will have unique business use cases and data assets that require protection."[35]

It is suggested that the execution approach be one of achieving objective milestones rather than technologies; with forgiveness on implementation failures, flexibility for timelines and iterative attempts using different solutions.

At this point, most security professionals may have questioned the phrase "forgiveness on failure" and perhaps have stopped reading in disgust. Note though, that failure with a particular technology should not equate to allowing a security breach. In fact, quite the opposite. The main point of this whitepaper, if it hasn't been made clear yet, is:

Zero-Trust must be implemented with the mindset of a "layered security" approach.

This means that a deploying any new solution should not disrupt or prematurely remove any existing security measures, at least until it has been proven to be ZT-worthy and production-ready.

With such a vague guidance to execution, how does an organization actually achieve tangible progress towards a Zero-Trust target state? The execution plan should be broken down into 6 major phases: **alignment, operations, technology, data, orchestration** and **maintenance**.

Aligned Collaboration – the Unwitting Team Member

The concepts of assets and importance of an asset inventory has been raised multiple times. What hasn't been discussed yet is: the relationship with other teams, the inventory of current projects, and the other strategic roadmaps executives have committed to.

Ideally, all in-flight projects must be congruent to the Zero-Trust overall strategy and align towards same target state.

In reality most projects may be aligned to a ZT (or other progressive security framework), need minor course corrections, or inconsequential to ZT. Inevitably, there will be some projects which may be counterproductive to the ZT target state. When alignment to the ZT initiative is impossible, there are several possible alternatives:

- **Reworking the outlier** in question to find alternatives that meet their needs. The costs for this change may be assessed to the ZT-initiative, but the cost of not doing this could be more. In one case, we found that a business unit was implementing a conflicting solution because they were not aware of the Zero-Trust initiative; but once apprised of the ZT roadmap, they quickly took on the ZT-centric solution.

- **Reworking the ZT design** and initiative to adjust and work around the outlier without affecting the totality of security. In some cases, this may be as simple as segmenting the outlier into its own sandbox, or it could be as complicated as building customized complex solutions.

- **Using executive arbitration** to decide the course of action. This method should always be saved as the final option, since it will be costly both politically as well as financially; and each party will never be satisfied with the mandated outcome.

This first hurdle might be the deal-breaker if there are too many projects out of alignment – which could also imply that the Zero-Trust design was too isolated from the reality of the existing infrastructure and culture of the organization.

Order of Operations – Perfecting the Recipe

Every good chef knows to test their recipes in small batches with trusted and honest test pools. This allows the chef to take constructive criticism and refine the recipe for the next level of delivery, eventually leading to mass distribution. This is known as the order of operations.

Edward Nash Yourdon defined a similar order of operations for structured programming, known as Yourdon Structured Method (YSM) back in the early 1970’s which was a precursor to object-oriented programming designed patterns in the 1980’s. Whether most developers know Yourdon or not, they are taught his methodology from the very start of their programming education.

This same method is applied to large system designs as well. In any large infrastructure project, of which Zero-Trust applies, there is the sandbox or testbed for verifying the isolated functionality of each individual solution and verification of APIs, the cyber lab where interconnectivity and integration testing are performed, and then the pilot. Each progression provides valuable input into subsequent iterations, minimizing the possibility of systemic failures.

The pilot program selects a willing, trusted, resilient and preferably non-critical (and unregulated) business unit to subject themselves to a small-scale migration to the new reference implementation. Existing business resiliency is imperative since there is the clear possibility of discovering new weaknesses, minor failures and iterative deployments.

As the merits of grouping were discussed previously, implementing a “pilot” is the idyllic use case for supporting the grouping concept. With successful grouping, multiple pilots of varying sizes and shapes can take place; incrementally spreading the footprint of Zero-Trust across the organization.

Defining the proper order of operations may not be as resource intensive as alignment, but it bears an explicit step towards success deployment of Zero-Trust.

Technology Flexibility – an Open Road with Guardrails

Flexibility and iterations have been the mainstay of this effort. Make no mistake, Zero-Trust will be an ongoing journey towards an ideal target state. The key to maintaining flexibility and supporting iterations is to implement modularity design without becoming prescriptive.

When defining an execution plan, we are more apt to present milestones using vendor-specific solutions because it is easier to understand and justify. However, the proper scribing of the plan should be etched with the completion of functional requirements, which may make the plan more complicated to read.

This seemingly odd planning tactic is recommended because the ZT journey will take years to cover the entire organization, and as such, new technologies and solutions will replace older ones – but the functional requirements should stay relatively stable.

Also, when a solution fails or a vendor states the end-of-life for a product, it will be easier to search for new vendors based on documented functional requirements instead of backtracking the features that the existing vendor provided.

There will never be a one-for-one solution replacement, but by maintaining the map of ZT requirements overlaid with the deployed solutions, akin to the Grigorof map[36] mentioned previously, finding a replacement becomes a much easier.

Sensor Data Harmonization – Different Instruments Playing the Same Song

The next hurdle in the execution plan will finding a way to get all the integral parts of the security framework – both incoming and incumbent technologies – to supply palatable data to the control plane policy information point (PIP).

The Health Information Technology for Economic and Clinical Health (HITECH) Act of 2009 required all medical providers to utilize electronic health records (EHR); but it never specified a standard for the data format. What was a cost-saving gesture actually turned out to increase healthcare costs, and the creation of two new businesses: (1) those that converted paper records into electronic records, usually proprietary in nature for vendor lock-in, and (2) integration companies that converted data in those proprietary formats into a second format for exchanging with other healthcare companies.

The Zero-Trust initiative must ensure that the same issue of proprietary lock-in doesn't occur with the various deployments and control plane implementation. Until there is a control plane data exchange standard, we are left to ensure that data is harmonized in a reusable way.

As security professionals, we (collectively) must make a concerted effort to build those standards for harmonized data exchange format for both ingestion of sensor data as well as policy decision data. If it is our responsibility to the security community to build a vendor-agnostic framework; if not us, then we are left to the vendors to do it for us.

Orchestration – The Moon Shot

The importance of the XACML control plane in the Zero-Trust paradigm is a testament to the broader concept of orchestration.

Orchestration is the underpinning for Zero-Trust success, as it dynamically defines the interrelationships between assets, sensors, controls and policies.

It is worth restating several concepts presented throughout this paper:

- What orchestration did for transforming virtualization into the Cloud, orchestration will transform security controls into Zero-Trust.

- Today, each security tool provides its own version of a control plane, posturing acceptance through its “universal” console, but at the same time, creating vendor lock-in implementing proprietary data formats and limiting integration.

- Within the next decade there will be consolidation amongst vendors resulting in two or three major independent control plane implementations, which all other security vendors will integrate to.

This state of affairs leaves organizations in a quandary:

- Can an organization build their own control plane, along with all the tedious data integrations, unproven decision models, and manually executed enforcement?

- Should the organization support multiple disparate vendor-specific control plane implementations, accepting the data disconnects, lack of contextual decision making, and automated policy enforcement; but with the possibility that different solutions may create conflicting rules/policies?

- Is there a balance whereby the target state can be designed to utilize existing vendor solutions – albeit accepting imperfections – but allow for a future where such deployments can be replaced with a ubiquitous control plane?

While there is no magic checklist that can provide how this can be done for every organization, it bears formal discussions to come to an explicit and documented plan of action.

Choosing to Build the Control Plane Components

We recommend that as much of the control plane functionality be implemented using serverless computing and/or containerization to be modular, ephemeral and scalable.

Purchasing Commercial Control Plane Components

We recommend that there is more benefit to consolidating vendors than trying to select best-of-breed in each category. Vendor consolidation reduces security risk for the following reasons:

Risk	Multiple Vendor Point Solutions	Consolidated Vendor Solution
Vendor Security	**Difficult to Protect:** • Disparate Data Stores • Encryption Key Management Nightmare • Multiple Access Points / Attack Surfaces • Multiple Exposures of Internal Directories	**Reduced Attack Surface:** • Single Data Store • Simplified Encryption Key Management • Single Access Control Point • Single Exposure of Internal Directories
Security Operations	**Inability to Meet Objectives:**	**Optimal Defense Operations:**

	• Disconnected Intelligence • Inadequate Reporting • Unmanageable Noise-to-Signal Ratio	• Shared Intelligence Across Solutions • Contextually Complete Reporting • Highly Efficient SOC Operations
Security Coverage	**Composite Topology:** • Gross Overlap = Wasted $$$ • Unknown Gaps = Immeasurable Exposure	**Comprehensive Topology:** • Tight Integration = Optimal ROI • Known Gaps = Manageable Exposure
Supply Chain	**Complex Vendor Management:** • Unaligned License Renewal Cadence • Multiple Support Teams	**Simple Vendor Management:** • Simplified License Renewal Process • Single Point-of-Contact

In the end, don’t let a purist model of perfection get in the way of building an effective security orchestration platform.

Maintenance – Care & Feeding

When contemplating any technology or security solution, an organization must take into consideration the up-front licensing and deployment costs as well as the recurring monthly maintenance costs. Maintenance is more than vendor support subscription fees; it includes internal resource hours.

An organization’s Zero-Trust initiative must also consider the maintenance costs across not only the vendor solutions employed, but the remediation and patching state for every asset.

Zero-Trust Takeaways

There is no easy way to attain Zero-Trust in your organization, there is no checklist to work from, there is no standard reference implementation that applies equally to every situation, and there is no vendor that can satisfy all your specific Zero-Trust requirements.

What we've attempted to do is build awareness into all the dependencies and characteristics you will need to define your Zero-Trust vision, design your target state infrastructure, and create a roadmap for migration to a better security posture.

Given all the uncertainty and challenges your organization will face when embarking on the Zero-Trust journey, it is important to remember the following about the ZT mindset:

- **Holistic** – not Wholesale – Change
- **Asset** is the Perimeter (not the User)
- Categorizing by **Actors** and **Conduits** helps build the right mix of controls
- **Orchestration** of Security Controls is central to operations
- **Journey of Maturity** from Static to Dynamic Security
- **Design to Requirements** not to Vendor Features

*Don't **START** with Zero-Trust … work your way **TO** Zero-Trust.*

Finally, it is strongly recommended that an independent cross-industry coalition be formed to manage the issues of lexicon, standardized data exchange, reference implementations and independent participation in the maturing of NIST SP 800-207 – similar to INCITS and ANSI. In addition, sub-groups (SIGs) should be created for specific industry verticals that may need to collaborate for ensuring full security coverage for multi-organizational transactions.

Recommended Reading

NIST SP 800-207 (draft)
https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-207-draft2.pdf

Return on Security Investment (ROSI): A Practical Quantitative Model
https://infosecwriters.com/text_resources/pdf/ROSI-Practical_Model.pdf

Distributed Immutable Ephemeral - New Paradigms for the Next Era of Security
https://www.rsaconference.com/industry-topics/webcast/35-new-paradigms-for-the-next-era-of-security

Jericho Forum Commandments
https://collaboration.opengroup.org/jericho/commandments_v1.2.pdf

Identifying Unintended Harms of Cybersecurity Countermeasures
https://www.cl.cam.ac.uk/~ytc36/Identifying_Unintended_Harms.pdf

Zero Trust Networks: Building Secure Systems in Untrusted Networks
https://www.amazon.com/Zero-Trust-Networks-Building-Untrusted/dp/1491962194/

BeyondCorp: A New Approach to Enterprise Security
https://research.google/pubs/pub43231/

eXtensible Access Control Markup Language (XACML) Version 3.0
http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html

SDP Specification 1.0
https://downloads.cloudsecurityalliance.org/initiatives/sdp/SDP_Specification_1.0.pdf

About the Author

J.C. Checco, C|CISO, CISSP, CSSLP, CCSK

J.C. Checco is Resident CISO leading the CISO Advisory Board for Financial Services at Proofpoint. Prior to this role J.C. was Senior Vice President for Bank of America's Security Research & Innovation team, a key contributor to the Financial Systemic Analysis & Resiliency Center (FSARC), and the inaugural participant in the DHS Loaned Executive Program at NCCIC (now CISA). He also served as CISO for BloombergBlack as well as Senior Information Security & Risk Advisor for Bloomberg LLP.

J.C. supports the information security community as President Emeritus of the NY Metro InfraGard Members Alliance (an FBI public/private partnership), member of the Strategic Advisory & Content Committee of the Wall Street Technology Association and co-founder of the annual NY Metro Joint Cyber Security Conference.

J.C. is well represented in the field of technology including patents in Unified Messaging and Keystroke Biometrics, as well as numerous pending patent applications in the fields of Gesture Biometrics, Blockchain Security, IoT Security, Zero Trust, and 5G Security. His experience encompasses emerging technology research and development at Bank of America Security Research & Innovation, NYNEX Science & Technology, Pitney Bowes Advanced Concepts & Technology, Advanced Technology Labs and IBM's T. J. Watson Research Center.

Follow, connect and read more from J.C. Checco at https://www.linkedin.com/in/checco.

About Proofpoint

Proofpoint is a publicly traded (PFPT) pure-play cybersecurity company based in Sunnyvale, California. It is our people-centric approach that makes is unique in the cybersecurity industry, and Proofpoint leads the market because of that focus. We protect many of the world's largest, industry-leading customers. Our customers include most of the Fortune 100, Fortune 1000, Global 2000 and thousands more worldwide.

Our deep security DNA is why we're a top cybersecurity company. We've sustained many years of leadership according to industry analysts—no one is close to that. We've appeared in four Gartner Magic Quadrants (MQ): Secure Email Gateway (now a Market Guide), Enterprise Information Archiving, Cloud Access Security Broker (CASB) and Security Awareness Computer-Based Training. Proofpoint has been in the upper right "Leaders" quadrant for several consecutive years.

In January 2020, Proofpoint received FedRAMP authorization for its core email security and archiving products. Proofpoint is committed to providing its state-of-the-practice capabilities to financial services in order to assist in protecting our critical economic infrastructure as well as bolstering consumer trust in the financial services cybersecurity ecosystem.

LEARN MORE

For more information, visit **proofpoint.com**.

proofpoint.

Endnotes & References

[1] https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-207-draft2.pdf

[2] https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-207-draft2.pdf

[3] https://www.nfpa.org/Public-Education/Staying-safe/Preparedness/Fire-Prevention-Week/About

[4] St. Louis Post-Dispatch (2020), “A look at some of the most notorious serial killers in the US since 1970”

[5] Beardsley (2013), “Security 101: Understanding the Common Layered Security Concept”

[6] https://collaboration.opengroup.org/jericho/commandments_v1.2.pdf

[7] https://www.globalidentityfoundation.org/downloads/Identity_30_Principles.pdf

[8] https://www.darkreading.com/threat-intelligence/soc-wins-and-losses/d/d-id/1338184

[9] Sonnenreich (2005), “Return on Security Investment (ROSI): A Practical Quantitative Model”

[10] Chua, Parkin, Edward, Oliveira, Schiffner, Tysonk and Hutchings (2019) “Identifying Unintended Harms of Cybersecurity Countermeasures”

[11] Yu (2019), “Distributed Immutable Ephemeral - New Paradigms for the Next Era of Security”

[12] Sidebar: A Google search showed ~200K references to the phrase “user is the perimeter”, ~19K references to “endpoint is the perimeter” and no results for “asset is the perimeter.” We’re here to change that.

[13] https://www.globalidentityfoundation.org/downloads/Identity_30_Principles.pdf

[14] https://attack.mitre.org/

[15] https://oasis-open.github.io/cti-documentation/stix/intro.html

[16] https://oasis-open.github.io/cti-documentation/taxii/intro.html

[17] https://web.mit.edu/kerberos/

[18] https://network-insight.net/2019/06/zero-trust-single-packet-authorization-passive-authorization/

[19] https://spiffe.io/

[20] Checco (2014), “Review of ICS/SCADA Risks”

[21] https://www.cisco.com/c/en/us/solutions/software-defined-networking/sdn-vs-nfv.html

[22] https://www.opennetworking.org/sdn-definition/

[23] https://www.cisco.com/c/en/us/solutions/software-defined-networking/sdn-vs-nfv.html

[24] http://www.waverleylabs.com/software-defined-network-sdn-or-software-defined-perimeter-sdp-whats-the-difference/

[25] Checco, Ogrinz (2019), “Robotic Process Automation: The Promise, the Patterns, and the Pitfalls”

[26] http://blog.icreon.us/advise/web-scraping-legality

[27] https://www.zestfinance.com/hubfs/Underwriting/Explainable-Machine-Learning-in-Credit.pdf

[28] Iannacone, Bridges (2018), “Quantifiable & Comparable Evaluations of Cyber Defensive Capabilities: A Survey & Novel, Unified Approach”

[29] Cybersecurity Insiders (2020), "Zero Trust Progress Report"

[30] https://www.ilo.org/wcmsp5/groups/public/---ed_protect/---protrav/---safework/documents/normativeinstrument/wcms_218475.pdf

[31] https://www.cdc.gov/disasters/chainsaws.html

[32] George V. Hulme (2003), “Security Goes the Distance,” Information Week

[33] https://www.managedsentinel.com/2019/05/23/cybersecurity-roadmap/

[34] https://www.complianceforge.com/reasons/hierarchical-cybersecurity-governance-framework/

[35] https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-207-draft2.pdf

[36] https://www.managedsentinel.com/2019/05/23/cybersecurity-roadmap/