Security Aspect of Government Enterprise Architecture Reference

Streamlining cybersecurity by identifying, understanding, and integrating tools within the government enterprise architecture.

Authors

Darren W Pulsipher

Anna Scott

The Security Aspect of the Government Enterprise Architecture Reference (GEAR) Logical View integrates protection mechanisms such as encryption, intrusion detection, and threat mitigation across all layers of the architecture. It ensures that all parts of the enterprise are safeguarded against evolving threats by embedding security measures into the Organizational, Process, and Physical Layers. This approach enables a resilient, policy-driven environment capable of addressing diverse threats across every architecture layer.

Table of Contents

Overview

The Government Enterprise Architecture Reference (GEAR) Logical View is a comprehensive blueprint that integrates and organizes complex systems within government enterprises. It encompasses key subsystems, including data, applications, platforms, infrastructure, and physical devices, forming a cohesive digital ecosystem. Central to this architecture are cross-cutting aspects that ensure consistency and alignment, among them, the Security and Identity Aspects. While distinct, these two aspects are deeply interdependent: the Identity Aspect governs authentication and digital identity management, while the Security Aspect enforces protection mechanisms such as encryption, intrusion detection, and threat mitigation. Together, they form a unified defense posture that spans all logical architecture components.

The Security Aspect is embedded across the GEAR architecture's Organizational, Process, and Physical Layers, ensuring that all parts of the enterprise are safeguarded against evolving threats. At the Organizational Layer, our proactive security policies and training programs are designed to prevent insider threats and external manipulation, providing a sense of reassurance. The Identity Aspect ensures only verified users can access critical systems, enhancing our security measures. In the Process Layer, security enforces controls like logging, auditing, risk management, and secure software development practices, whereas identity services ensure traceability and role-based access alignment. In the Physical Layer, the Security Aspect introduces protections such as surveillance, access controls, and secure hardware configurations to prevent unauthorized physical access, reinforced by identity mechanisms like biometric verification and badge authentication.

The GEAR Logical View is designed to be modular and adaptable, allowing organizations to specialize and evolve each area independently. While maintaining tight integration, this separation ensures that the Security and Identity Aspects work in concert to protect the enterprise. The Security Aspect provides the defenses, while the Identity Aspect ensures those defenses are only accessible to verified and authorized entities. This approach enables a resilient, policy-driven environment capable of addressing diverse threats across every architecture layer—organizational, procedural, and physical.

The Security Aspect in the GEAR architecture is structured into six interdependent sub-aspects, each addressing a critical security dimension. These sub-aspects range from foundational trust principles and hardened infrastructure to proactive threat detection and resilient recovery. This zero-trust architecture-aligned approach to safeguarding infrastructure, data, and services weaves security throughout the architecture, from design to operations, providing the necessary controls to maintain confidentiality, integrity, and availability across dynamic and distributed environments.

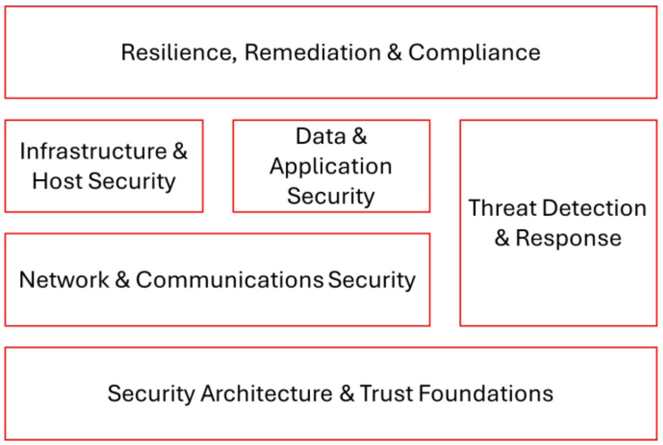
Resilience, Remediation & Compliance
Infrastructure & Host Security
Data & Application Security
Threat Detection & Response
Network & Communications Security
Security Architecture & Trust Foundations

Figure 1 Security Aspect High Level

Architecture & Trust Foundations

This foundational layer establishes the principles of security by design and zero-trust architecture. It defines the core building blocks that enable trust, such as product assurance, secure boot, attestation, and trusted execution environments (TEEs). By anchoring the architecture in verifiable integrity and provenance, this layer ensures that every component operates within a trusted and measurable framework, from hardware to firmware to applications/workloads.

Product Security Assurance / Secure Supply Chain

A known healthy state for all hardware (HW) is foundational to all security and identity requirements. Even the best-designed and executed security architecture can be undermined by HW/firmware vulnerabilities, bugs, and physical hardware alteration during manufacturing. HW providers typically have their own product security assurance programs, where they proactively search for issues and remediate them before they become commonly known and exploited. [1], [2], [3]. They also have their secure supply chain best practices and manufacturing controls to ensure that their products meet customer expectations and minimize the chance of counterfeiting or product tampering. Given the recent focus on supply chain and manufacturer location, the tools available for unit-level traceability and advanced provenance/surveillance technologies are constantly improving. They can be leveraged to provide better visibility of HW details for organizations.

Zero Trust Architecture

Zero Trust is the current approach for protecting an organization's digital infrastructure. It is focused on assuming that all computing operates in a hostile environment where devices, data, and user identity are constantly attacked. This means that throughout a digital architecture, the zero trust method is "never trust, always verify," so devices, users, networks, and applications/workloads are challenged to establish their credentials and operating permissions. Data must be protected in all states (at rest, in transit, and in use) [4].

The main zero trust principles are:

1. Assume no implicit or explicit trusted zone in networks.
2. Strictly enforce identity-based authentication and authorization for all connections and access to infrastructure, data, and services.
3. Strictly enforce machine-to-machine (M2M) authentication and authorization for communication between servers and applications.
4. Generate risk profiles by monitoring and assessing user and device behaviors in real time to authorize users and devices to access resources.
5. Encrypt all sensitive data in transit and at rest.
6. Continuously monitor, collect, store, and analyze all events to assess compliance with security policies.
7. Centralize policy management and distribution.

As mentioned above, zero-trust architectures assume that the hardware and software begin with a known healthy state. Once established, the organization must develop its risk profile by performing a risk assessment. This step is critical to defining where to focus resources and capabilities and defining policies so that the zero-trust architecture is crafted to meet the organization's security needs. [5].

Root of Trust

Modern computing platforms increasingly incorporate hardware-based security technologies to establish a foundation of trust within the execution environment. These technologies typically include measured launch, protected execution, and hardware-enforced isolation. They are designed to defend against software-based attacks and protect the confidentiality and integrity of data created, stored, or processed within the system.

These technologies enable applications to execute within isolated, secure environments, reducing the risk of compromise from other software components operating on the same platform. Hardware-rooted security mechanisms are essential for building a trustworthy foundation, supporting secure application execution, and safeguarding critical data and system operations.

Secure Boot & Attestation

Secure Boot and Attestation are foundational in establishing trust within the software and hardware stack, ensuring that systems start and operate only with verified and authorized code. Secure Boot is a security standard that verifies that a device boots using only firmware and software that are cryptographically signed by trusted authorities. During the boot process, the system's firmware checks the signatures of the bootloader, operating system, and other critical components, preventing the execution of malicious or tampered code. This process helps ensure attackers cannot compromise a device before fully loading the operating system and security tools. At higher levels of the software stack, similar principles are applied to container platforms and hypervisors, where signed and validated images are required before runtime, maintaining the integrity of workloads from the ground up.

Attestation complements Secure Boot by providing mechanisms for systems to prove their integrity to other systems or central authorities. A device can generate cryptographic proofs through remote attestation that it is running trusted software configurations, which a trusted attestation server can verify. Modern attestation processes often leverage hardware-based roots of trust, such as Trusted Platform Modules (TPMs) or secure enclaves to anchor the security claims. In cloud-native and edge environments, attestation ensures containers, virtual machines, and IoT devices run authentic, approved code before accessing sensitive data or critical network segments. By combining Secure Boot and Attestation, organizations create a chain of trust that extends from the hardware level through the entire software lifecycle, significantly raising the barrier against supply chain and runtime attacks.

Trusted Execution Environments (TEE) / Confidential Computing

Trusted execution environments are critical for protecting data in use in shared environments. In the absence of TEEs, the use of data and algorithms/models is riskier or must be conducted in an isolated on-prem architecture that, of necessity, cannot support data sharing. TEEs form the basis of confidential computing.

Given the broad adoption of public clouds and the security challenges posed by nosy neighbors, hackers, and admin/tenant access in these multi-tenant environments, TEEs are especially critical for protecting sensitive data. The rise of AI and the need for data from multiple organizations for model/algorithm development have made TEEs even more crucial. [6].

TEEs are all hardware-based, and they require, at a minimum, that the TEE hardware manufacturer and the code running in the TEE be trusted. Once established, they allow organizations to control and protect their sensitive data and models/algorithms by ensuring that only authorized users have access and preventing tampering, viewing, or theft. TEEs can also allow controlled use of 3rd party data while maintaining complete confidentiality, allowing large-scale collaboration without risking inadvertent sharing of regulated or sensitive data. [7].

TEEs require a method of attestation of the HW and SW components within a TEE and they must be used with encryption that protects data at rest and in transit.

To improve security even more, applications can be in secure enclave, which is a secure, isolated portion of memory within a TEE [8] .

Infrastructure & Host Security

Focused on the physical and virtual layers, this sub-aspect enforces hardening and protective measures across servers, containers, and edge devices. It includes runtime protections, host-based firewalls, and endpoint detection and response (EDR) to guard against compromise. By securing the compute fabric itself, this layer mitigates risk at the level where attackers often seek persistence and control.

System Hardening

System hardening at the software and logical layer reduces the attack surface by configuring and securing software components to minimize vulnerabilities. This includes turning off unnecessary services, removing unused software packages, and applying secure configuration settings for all elements within the stack. Automated configuration management tools like Ansible, Chef, or Puppet are often leveraged to ensure system settings comply with security baselines (e.g., CIS benchmarks). They are consistently applied across development, staging, and production environments. Additionally, patch management systems ensure that all known vulnerabilities are addressed promptly through software patches or updates, especially for critical services or systems exposed to the internet.

Furthermore, secure software development practices (e.g., code review, secure coding standards) and static analysis tools during the development lifecycle help harden applications from the outset. Components like web application firewalls (WAFs) or runtime application self-protection (RASP) are also integrated at the software layer to actively prevent common attack vectors, such as SQL injection or cross-site scripting (XSS), from exploiting vulnerabilities in the application code. By fortifying the software stack through systematic hardening, organizations can mitigate the risks of exploitation and unauthorized access at both the application and service levels.

Runtime Protection

In the logical stack, runtime protection focuses on maintaining secure execution environments for applications, containers, and virtual machines during runtime. This involves deploying runtime application self-protection (RASP), virtual patching, and integrated anomaly detection. RASP solutions provide an additional layer of protection by embedding security directly within applications, where they monitor and block attack attempts in real-time, before they can exploit vulnerabilities in the code. Similarly, virtual patching tools can protect systems from known vulnerabilities without requiring immediate patching, giving security teams time to test and deploy updates without exposing systems to risks.

For containerized environments, runtime protection is integrated into platforms like Kubernetes using tools such as Falco or Sysdig, which monitor container behavior and enforce security policies on containerized workloads. These tools help detect unauthorized processes, privilege escalations, or file system changes that could indicate malicious activity. Additionally, integrated threat intelligence provides real-time feeds on emerging attack tactics, enabling quick response to threats in the software stack's runtime. This proactive approach to runtime security ensures that even if a vulnerability exists in the code, the execution environment is continuously monitored and protected from exploitation.

Host-based Firewalls & EDR

Within the software/logical stack, host-based firewalls and Endpoint Detection and Response (EDR) systems work together to safeguard individual systems and endpoints, such as servers, virtual machines, and containers. Host-based firewalls enforce network-level security by filtering inbound and outbound traffic on the host machine, preventing unauthorized access to critical services and blocking malicious traffic from reaching vulnerable ports. These firewalls are typically tightly integrated with the host's operating system and can be configured to apply specific rules based on application requirements, ensuring minimal exposure to network threats.

EDR solutions provide deeper visibility into endpoint activities, detecting and responding to potential threats that bypass traditional network defenses. They continuously monitor for suspicious activities, including unauthorized system access, malware infections, or abnormal application behavior. EDR systems can identify known and unknown attacks by combining real-time behavioral analysis with threat intelligence. These systems also support rapid incident response, enabling the isolation of compromised endpoints, the execution of forensic analysis, and the automation of remediation tasks, thus helping organizations contain threats before they can escalate. Integrating these protections within the software stack ensures that endpoints remain secure, even as adversaries increasingly target them for lateral movement and persistence.

Edge & IoT Security

Edge and IoT security in the software/logical stack ensures that devices and systems operating at the network's edge are protected against physical and cyber threats. IoT devices and edge computing platforms often present unique security challenges due to their distributed nature, limited compute resources, and connectivity to sensitive networks. To mitigate risks, security tools such as network segmentation and device-level access controls are implemented to ensure that only authorized devices can communicate with critical infrastructure or other devices on the network. Encryption protects data in transit between edge devices and central systems, ensuring attackers do not intercept sensitive information.

On the software layer, security policies are enforced through device management platforms, which allow administrators to deploy firmware updates, monitor device health, and ensure compliance with security standards. In edge computing environments, where computation is often performed close to the data source, containerization and microservice architectures help maintain the security of applications running at the edge. Runtime protection tools are deployed to monitor the behavior of applications at the edge, detect anomalies, and mitigate potential risks without requiring constant communication with centralized systems. These multi-layered protections provide robust defense against attacks targeting edge devices and enable secure, scalable IoT and edge computing solutions across the broader software stack.

Network & Communications Security

This sub-aspect protects the confidentiality and integrity of data in motion. It prevents unauthorized access and lateral movement by leveraging technologies like microsegmentation, SDN-based controls, and pervasive encryption for east-west and north-south traffic. DDoS protection and control plane security are also integral, especially in cloud-native and distributed deployments. [9].

East-West & North-South Encryption

East-West and North-South encryption are foundational to protecting data in motion within modern IT environments. North-South encryption secures traffic entering and exiting a data center or cloud environment—typically the focus of traditional perimeter defenses—while East-West encryption targets traffic moving laterally within the same network, such as between servers or virtual machines inside a data center. [10]. As organizations increasingly adopt distributed and cloud-native architectures, the volume of East-West traffic has grown significantly, sometimes accounting for up to 70% of all data center traffic. This shift has exposed a critical security gap, as most legacy security tools and firewalls focus on the perimeter and do not inspect or protect internal lateral flows.

Encrypting East-West and North-South traffic is essential for maintaining the confidentiality and integrity of sensitive data as it traverses the network. By applying pervasive encryption, organizations ensure that even if attackers penetrate the perimeter, the data moving internally remains protected and unreadable without the necessary keys. This approach is especially vital in environments with highly dynamic workloads and data, such as cloud and virtualized data centers. Technologies like SSL/TLS, IPsec, and emerging solutions for intra-data center encryption are increasingly being deployed to address these needs.

The benefits of comprehensive encryption extend beyond compliance and data privacy. By encrypting all data in motion, organizations can effectively hinder threat actors' lateral movement, making it significantly more difficult for attackers to access or exfiltrate sensitive information after breaching initial defenses. This layered approach to encryption and other security controls forms a robust defense-in-depth strategy well-suited for modern IT infrastructures' complex, distributed nature.

Microsegmentation

Microsegmentation is a security strategy that divides networks into granular, isolated segments, each with tailored security policies. [11]. Unlike traditional network segmentation, which might separate networks coarsely (such as by department or application), microsegmentation operates at a much finer granularity, isolating individual workloads, virtual machines, or containers. This approach is particularly effective in preventing the lateral movement of threats within a network, as attackers who compromise one segment cannot easily move to others. [12].

Microsegmentation can be implemented through various methods, including agent-based solutions that enforce policies at the host level, network-based controls leveraging SDN or virtual switches, and native cloud controls provided by major cloud service providers. These solutions provide visibility into all internal network traffic and allow dynamic adaptation as workloads and environments change. For example, in cloud-native environments where workloads are ephemeral and highly dynamic, microsegmentation policies can automatically adjust in real time to maintain security as resources are created or destroyed.

Organizations adopting micro-segmentation benefit from a reduced attack surface, improved breach containment, and stronger regulatory compliance [9]. By isolating critical workloads and applying least-privilege access principles, micro-segmentation not only limits the potential impact of a breach but also simplifies policy management and incident response. This makes it an essential component of Zero Trust security models and a best practice for securing both on-premises and cloud-based infrastructures.

SDN Security

Software-Defined Networking (SDN) introduces a programmable and centralized approach to managing network traffic, enabling more agile and responsive security controls. SDN-based security allows organizations to dynamically apply policies, segment networks, and monitor traffic flows in real time, all from a central controller[1][4]. This is particularly valuable in environments with high volumes of East-West and North-South traffic, as SDN can enforce encryption, microsegmentation, and access controls consistently across physical and virtual network layers.

One key advantage of SDN security is its ability to provide granular visibility and control over network traffic, down to the level of individual workloads or applications. [9]. This enables organizations to quickly detect and respond to anomalies, enforce least-privilege access, and limit the spread of threats within the network. SDN also facilitates the implementation of dynamic security policies that can adapt to changes in the network environment, such as adding or removing virtual machines, containers, or cloud resources.

SDN security is often integrated with other technologies, such as micro-segmentation and pervasive encryption, to provide a comprehensive defense-in-depth strategy. For example, SDN controllers can orchestrate the deployment of encryption for East-West and North-South traffic, coordinate segmentation policies, and ensure that security controls are uniformly applied across hybrid and multi-cloud environments. This level of automation and integration is essential for securing modern, distributed IT infrastructures where manual policy management would be impractical and error-prone.

DDoS Protection

Distributed Denial of Service (DDoS) protection is critical to network security, especially in cloud-native and distributed deployments where services are exposed to the public internet and can be targeted by large-scale attacks. DDoS attacks aim to overwhelm network resources, disrupt service availability, and potentially serve as a smokescreen for more targeted intrusions. Adequate DDoS protection involves a combination of detection, mitigation, and resilience strategies that can scale to counter evolving threats.

Modern DDoS protection solutions leverage various techniques, including traffic analysis, rate limiting, and behavioral analytics, to identify and block malicious traffic before it can impact critical services. In cloud environments, DDoS protection is often integrated with SDN-based controls and cloud provider security services to ensure rapid detection and automated response. This integration allows organizations to reroute or filter traffic dynamically, absorb attack volumes, and maintain service availability even under sustained assault.

Beyond protecting against volumetric attacks, DDoS protection also plays a role in safeguarding the control plane—the management layer that orchestrates network and application resources. Attacks targeting the control plane can disrupt user-facing services and the underlying infrastructure, making robust protection essential for maintaining operational continuity in distributed and cloud-native environments. By combining DDoS protection with encryption, microsegmentation, and SDN security, organizations can build a resilient security posture capable of withstanding external and internal threats.

Data & Application Security

Here, security is embedded into the application lifecycle and data handling practices. It includes encryption across all data states, robust secrets management, and secure development practices within DevSecOps workflows. By safeguarding applications and the data they manage, this layer helps prevent breaches at one of the most targeted layers of modern architectures.

Encryption (at rest, in transit)

Encryption for data at rest (in storage) and in transit (on the network) has been a fundamental security requirement for decades. However, the advent of quantum computers is compelling the upgrade of the algorithms being used for encryption. No one can predict precisely when quantum computers will be powerful enough to fulfill their potential, but the NSA guidance is for all systems to be quantum-ready by 2035 [13].

Encryption embedded in hardware comes in three main forms:

Firmware and SW signing

Firmware and software is signed for authenticity by the developer. The threat is that an attacker, equipped with the manufacturer's signing key which was broken by a quantum computer, could sign malicious firmware or software. The software would be viewed as genuine since it was signed with the manufacturer's key. NIST has specified new algorithms for post-quantum resistance (PQR).

Symmetric

Symmetric (single key) encryption is foundational to data protection. Disk, file and database encryption is almost always done using symmetric algorithms. The critical secret exchanged with public/private key encryption is often a shared symmetric key. Fortunately, today's symmetric and hashing algorithms hold up against the quantum threat with sufficiently large key size. AES with 256-bit or stronger keys and SHA-384 or stronger are the symmetric encryption and hashing algorithms standardized by NIST for PQR.

Asymmetric

Asymmetric (public / private key) encryption are used to protect transmission of data or secrets across a network or other connections, including web browsing, cloud services access, VPNs, and other use cases. Unfortunately, Shor's algorithm and a sufficiently powerful quantum computer could break today's asymmetric encryption algorithms, including RSA, Elliptic Curve, and Diffie-Hellman. New algorithms are needed for asymmetric encryption in the quantum era. NIST selected ML-KEM (formerly CRYSTALS Kyber) and ML-DSA (formerly CRYSTALS Dilithium) [13] algorithms, standardizing them in August 2024. ML-KEM is used for key establishment between parties across an untrusted connection, and ML-DSA is used for signing and authenticating digital signatures.

Secure Software Supply Chain

In a layered system architecture, security ensures the integrity and trustworthiness of all components within the software stack. The secure supply chain capability specifically addresses the protection of source code, dependencies, build processes, and deployment artifacts, ensuring that every element used to construct and operate the software stack is verified, authentic, and free from tampering.

This security service leverages automated tools, hardware-backed attestation mechanisms, and tightly integrated verification controls within the software stack.

Secure Supply Chain Tools and Technologies in the Software Stack

In today's complex digital landscape, ensuring the security and integrity of software systems has become paramount. As organizations increasingly adopt advanced technologies and distributed architectures, the need for robust security measures has never been greater. Organizations can effectively safeguard their assets and maintain operational continuity by understanding and implementing these practices.

- Version Control and Code Signing – Source code and configuration files are stored in signed, auditable repositories. Comments and tags are signed to prevent unauthorized changes and provide non-repudiation.
- Software Bill of Materials (SBOM)—Each software component and dependency in the stack is documented with a machine-readable SBOM, which enables dependency tracking, compliance checks, and incident response.
- Dependency and Package Scanning – Automatically scans and validates all dependencies (open-source, commercial, or internal) integrated into the stack, identifying and remediating known vulnerabilities before promotion to production.
- Secure Build Pipelines and Artifact Provenance — Continuous Integration systems enforce isolated, reproducible builds. Artifacts are cryptographically signed and traced back to the source, preserving provenance and ensuring trust.
- Container and Image Security – Containers and other binary artifacts are signed and verified during promotion and deployment. Trusted registries enforce signature validation.
- Runtime and Deployment Verification — Policy engines ensure that only signed and verified software components are deployed into runtime environments.
- Hardware-Backed Integrity and Attestation – Attestation frameworks validate the identity and integrity of software and hardware at boot or runtime, enabling trusted execution environments for critical services in the stack.
- Immutable Infrastructure & IaC Verification — Infrastructure as Code (IaC) is scanned and validated as part of the build process, ensuring that infrastructure configurations do not introduce vulnerabilities or drift.

Integrating advanced security measures across all stages of the software development lifecycle is essential for protecting against external and internal threats. From DDoS protection and quantum-resistant encryption to secure supply chains and automated security testing, each layer plays a critical role in maintaining the integrity and trustworthiness of modern software systems. By adopting a holistic approach to security, organizations can build a resilient posture that protects their assets and fosters trust and reliability in their services and products.

DevSecOps & CI/CD Security

Early and Continuous Security Integration

Security starts at the planning and architecture phases. Teams perform threat modeling and risk assessments before a line of code is written. Security plugins in IDEs (e.g., GitHub Advanced Security, JetBrains Security Analysis) and automated checks in version control systems (e.g., Git hooks, pre-commit checks) provide instant feedback and enforce secure coding practices.

Automated Security Testing in CI/CD

Security testing is embedded into the CI/CD pipeline using tools like:

- SAST (Static Application Security Testing)- Analyzes source code or binaries for vulnerabilities without executing the programs.
- DAST (Dynamic Application Security Testing)- Tests running applications for vulnerabilities by simulating external attacks.
- Dependency Scanning- Identifies and mitigates risks arising from third-party libraries and dependencies.
- IaC Scanning- Examines infrastructure-as-code scripts to ensure secure configurations and prevent drift.

These tools automatically detect and block known vulnerabilities in code, third-party libraries, containers, and infrastructure-as-code before they reach production.

Integration of Trusted Execution Environments

For high-assurance workloads, TEE introduce hardware-enforced isolation for sensitive code and data. These environments protect against host-level attacks, even from compromised kernels or hypervisors. TEE in CI/CD systems are best used in the following use cases:

- **Sensitive services** (e.g., key management and authentication microservices) run inside enclaves, ensuring data is encrypted when used.
- **Artifact signing and attestation** are performed within enclaves during the build process, enhancing software supply chain trust.
- **Remote attestation** verifies the enclave's integrity before sensitive transactions or access are allowed.

Threat Detection & Response

This layer provides real-time monitoring and analysis of system behavior to combat evolving threats. It includes tools like EDR/XDR, user and entity behavior analytics (UEBA), and deception technologies to uncover known and novel attacks. Integrating threat intelligence and automated response mechanisms enables faster detection, containment, and learning from incidents.

EDR/XDR

- EDR tools continuously monitor endpoints (servers, containers, VMs) for suspicious behavior, unauthorized access, and exploitation patterns.
- XDR expands this visibility across multiple domains—network, identity, email, cloud workloads—aggregating data from disparate tools to provide centralized threat correlation, root cause analysis, and response orchestration.
- These tools integrate with CI/CD environments and cloud-native platforms to detect threats close to the workload.

Threat Intelligence & Analytics

- Real-time threat intelligence feeds (from commercial, open-source, and private sources) are integrated to provide up-to-date information on known attack vectors, indicators of compromise (IOCs), and emerging tactics.
- Machine learning-based analytics detect anomalies that diverge from baseline system or

application behavior, surfacing potential threats without relying solely on signatures.

- Stack-integrated threat intelligence also informs access decisions and runtime policies (e.g., microsegmentation, firewall rules).

UEBA & Insider Threat Detection

- UEBA tools profile users, service accounts, and machine identities, detecting deviations from established behavior such as abnormal access times, data exfiltration attempts, or privilege escalation.
- UEBA is especially useful for insider threat detection. It correlates identity, behavior, and access context across the software stack to uncover subtle threats.

Deception Technologies (Honeypots, Canaries)

- Deception tools such as honeypots, canary tokens, and fake credentials are embedded into the environment to lure adversaries into controlled traps.
- These elements serve as high-fidelity alerts—any interaction with them is anomalous by design, triggering immediate investigation or automated containment.
- Some modern platforms integrate deception directly into the CI/CD or service mesh layer to detect lateral movement attempts within microservice environments.

Automated Response and Feedback Loop

- Detected threats can trigger automated actions such as isolating containers, revoking credentials, rolling back deployments, or opening incident response tickets.
- Observed attack data feeds into detection models, improving accuracy and resilience against recurring attack patterns.
- Integration with SIEM/SOAR platforms enables scale orchestration of complex response playbooks.

Resilience, Remediation & Compliance

This final layer ensures systems can recover from compromise while meeting regulatory and operational mandates. It covers everything from automated recovery playbooks and audit logging to security chaos engineering and SBOMs. It promotes continuous assurance and readiness, ensuring that the architecture prevents attacks and can respond and adapt when they occur.

Remediation & Recovery Playbooks

- Predefined, tested incident response and recovery playbooks are embedded into orchestration layers, enabling automated rollback, failover, and service restoration.
- These playbooks often trigger based on alert thresholds or behavioral detections (e.g., from XDR or UEBA systems) and are integrated with CI/CD and container orchestration platforms (e.g., Kubernetes).

Examples include revoking compromised credentials, re-deploying known-good containers, or spinning up secure backup instances.

Security Chaos Engineering

- Inspired by principles of chaos engineering, this involves deliberately injecting faults, attacks, or misconfigurations into systems in a controlled way to test their resilience.
- This method helps validate that security controls (e.g., rate limiting, failover, alerting) behave correctly under duress.
- It also simulates attack scenarios (e.g., credential theft, lateral movement) in staging environments to test incident detection and response efficacy.

Audit Logging & Compliance

- Comprehensive audit trails capture security-relevant events across the stack: code commits, build changes, deployment actions, authentication logs, and runtime behavior.
- These logs are cryptographically signed and immutable, ensuring integrity for forensic analysis, compliance audits, and incident reviews.

- Integration with compliance frameworks (e.g., NIST 800-53, FedRAMP, ISO 27001, HIPAA) enables organizations to demonstrate control maturity and pass audits efficiently.

Software Bill of Materials (SBOM)

- A complete and traceable Software Bill of Materials is generated and maintained for every build artifact, including dependencies, libraries, and configuration files.
- SBOMs are signed, stored, and linked with build metadata, enabling artifact provenance, vulnerability scanning, and compliance validation.
- This layer also enforces controls like signed builds, verified registries, and reproducible builds, ensuring end-to-end trust across the stack.
- Hardware components and firmware dependencies are increasingly tracked alongside software in modern SBOMs to align with emerging national security mandates.

Using the Security Aspect in GEAR

The Security Aspect interweaves with each subsystem of the GEAR architecture to ensure an end-to-end, defense-in-depth security posture. This mapping outlines how the security sub-aspects integrate with and enhance the resilience of each major architectural layer.

GEAR Subsystem	Security Aspect Mapping
Application Layer	Secure Software Supply Chain, DevSecOps integration, Application/Data Security, Root of Trust for applications, Encryption at rest and in transit, Runtime protection.
Service Management Layer	EDR/XDR monitoring, Threat Intelligence, Automated Incident Response (SOAR), UEBA analytics, Zero Trust enforcement, Audit logging, Policy enforcement for service controls.
Distributed Information Management Layer	Encryption for East-West/North-South data flows, Microsegmentation, SDN security policies, Secure multi-party computation, TEEs, and confidential computing.
Software-Defined Infrastructure Layer	Secure Boot, Hardware Root of Trust, Attestation, Dynamic security policy orchestration, SDN-based microsegmentation, TPM-backed workload integrity validation.
Physical Layer	Secure Supply Chain, Hardware assurance, Physical security controls (access, surveillance, facility design), Hardware Root of Trust, Intrusion detection for physical assets.
Identity Aspect	Federated Identity, Risk-based Authentication/Authorization, Zero Trust enforcement, Behavioral analytics, Identity-driven access control across all layers.

References

[1] "Product Security Assurance," 2025. [Online]. Available: https://www.intel.com/content/www/us/en/security/product-security-assurance.html.

[2] "Product Security," 2025. [Online]. Available: https://www.amd.com/en/resources/product-security.html.

[3] "Product Security Incident Response Team (PSIRT) Policies," 2025. [Online]. Available: https://www.nvidia.com/en-us/product-security/psirt-policies/.

[4] "Zero Trust Reference Architecture," 2025. [Online]. Available: https://dodcio.defense.gov/Portals/0/Documents/Library/(UZT_RA_v2.0(U_Sep22.pdf)).

[5] "Zero Trust Maturity Model Version 2," 2025. [Online]. Available: https://www.cisa.gov/sites/default/files/2023-04/CISA_Zero_Trust_Maturity_Model_Version_2_508c.pdf.

[6] "Confidential Computing," 2025. [Online]. Available: https://confidentialcomputing.io/.

[7] "The Case for Confidential Computing," 2025. [Online]. Available: https://www.linuxfoundation.org/hubfs/Research%20Reports/TheCaseforConfidentialComputing_071124.pdf?hsLang=en.

[8] "Intel® Trust Domain Extensions (Intel® TDX)," 2025. [Online]. Available: https://www.intel.com/content/www/us/en/products/docs/accelerator-engines/trust-domain-extensions.html.

[9] "What is Microsegmentation," 2025. [Online]. Available: https://www.paloaltonetworks.com/cyberpedia/what-is-microsegmentation.

[10] "North-South and East-West Network Security," 2025. [Online]. Available: https://www.linkedin.com/pulse/north-south-east-west-network-security-etienne-liebetrau.

[11] "Microsegmentation," 2025. [Online]. Available: https://www.illumio.com/ja/cybersecurity-101/microsegmentation.

[12] "Why Micro-Segmentation Works So Well in the Cloud," 2025. [Online]. Available: https://www.hillstonenet.com/blog/why-micro-segmentation-works-so-well-in-the-cloud/.

[13] "Commercial National Security Algorithm Suite 2.0 FAQ," 2025. [Online]. Available: https://media.defense.gov/2022/Sep/07/2003071836/-1/-1/0/CSI_CNSA_2.0_FAQ_.PDF.

[14] "Intel® Trusted Execution Technology (Intel® TXT)," 2025. [Online]. Available: https://www.intel.com/content/www/us/en/support/articles/000025873/processors.html.