NIST SPECIAL PUBLICATION 1800-37

Addressing Visibility Challenges with TLS 1.3 within the Enterprise
High-Level Document

Bill Newhouse
Murugiah Souppaya
David Cooper
Tim Polk*
National Institute of Standards

William Barker
Stratvia LLC

Karen Scarfone
Scarfone Cybersecurity

John Kent
Julian Sexton
Michael Dimond
Josh Klosterman
Ryan Williams
The Mitre Corporation

David Wells
Johann Tonsing
Mira Security

Sean Turner
sn3rd

Patrick Kelsey
Not for Radio

Russ Housley
Vigil Security LLC

Tim Cahill
JPMorgan Chase & Company

Muralidharan Palanisamy
AppViewX

Dung Lam
F5

Paul Barrett
Ray Jones
Sandeep Jha
Netscout

Steven Fenter
Jake Wills
US Bank Corporation

Jane Gilbert
D'Nan Godfrey
Thales TCT

Dean Cocklin
Avesta Hojjati
DigiCert

*Contributed while a NIST Employee

September 2025

FINAL

This publication is available free of charge from
https://www.nccoe.nist.gov/addressing-visibility-challenges-tls-13

NIST SPECIAL PUBLICATION 1800-37

Addressing Visibility Challenges with TLS 1.3 within the Enterprise High-Level Document

Bill Newhouse
Murugiah Souppaya
David Cooper
Tim Polk*
National Institute of Standards

William Barker
Stratvia LLC

Karen Scarfone
Scarfone Cybersecurity

John Kent
Julian Sexton
Michael Dimond
Josh Klosterman
Ryan Williams
The Mitre Corporation

David Wells

Johann Tonsing
Mira Security

Sean Turner
sn3rd

Patrick Kelsey
Not for Radio

Russ Housley
Vigil Security LLC

Tim Cahill
JPMorgan Chase & Company

Muralidharan Palanisamy
AppViewX

Dung Lam
F5

Paul Barrett
Ray Jones
Sandeep Jha
Netscout

Steven Fenter
Jake Wills
US Bank Corporation

Jane Gilbert
D'Nan Godfrey
Thales TCT

Dean Cocklin
Avesta Hojjati
DigiCert

*Contributed while a NIST Employee

Final
September 2025

U.S. Department of Commerce
Howard Lutnick, Secretary

National Institute of Standards and Technology
Craig Burkhardt, Acting Under Secretary of Commerce for Standards and Technology and Acting NIST Director

DISCLAIMER

Certain commercial entities, equipment, products, or materials may be identified by name or company logo or other insignia in order to acknowledge their participation in this collaboration or to describe an experimental procedure or concept adequately. Such identification is not intended to imply special status or relationship with NIST or recommendation or endorsement by NIST or NCCoE; neither is it intended to imply that the entities, equipment, products, or materials are necessarily the best available for the purpose.

While NIST and the NCCoE address goals of improving management of cybersecurity and privacy risk through outreach and application of standards and best practices, it is the stakeholder's responsibility to fully perform a risk assessment to include the current threat, vulnerabilities, likelihood of a compromise, and the impact should the threat be realized before adopting cybersecurity measures such as this recommendation.

National Institute of Standards and Technology Special Publication 1800-37, Natl. Inst. Stand. Technol. Spec. Publ. 1800-37, 63 pages, (September 2025), CODEN: NSPUE2

NATIONAL CYBERSECURITY CENTER OF EXCELLENCE

The National Cybersecurity Center of Excellence (NCCoE), a part of the National Institute of Standards and Technology (NIST), is a collaborative hub where industry organizations, government agencies, and academic institutions work together to address businesses' most pressing cybersecurity issues. This public-private partnership enables the creation of practical cybersecurity solutions for specific industries, as well as for broad, cross-sector technology challenges. Through consortia under Cooperative Research and Development Agreements (CRADAs), including technology partners—from Fortune 50 market leaders to smaller companies specializing in information technology security—the NCCoE applies standards and best practices to develop modular, adaptable example cybersecurity solutions using commercially available technology. The NCCoE documents these example solutions in the NIST Special Publication 1800 series, which maps capabilities to the NIST Cybersecurity Framework and details the steps needed for another entity to re-create the example solution. The NCCoE was established in 2012 by NIST in partnership with the State of Maryland and Montgomery County, Maryland.

To learn more about the NCCoE, visit https://www.nccoe.nist.gov/. To learn more about NIST, visit
https://www.nist.gov.

NIST CYBERSECURITY PRACTICE GUIDES

NIST Cybersecurity Practice Guides (Special Publication 1800 series) target specific cybersecurity challenges in the public and private sectors. They are practical, user-friendly guides that facilitate the adoption of standards-based approaches to cybersecurity. They show members of the information security community how to implement example solutions that help them align with relevant standards and best practices, and provide users with the materials lists, configuration files, and other information they need to implement a similar approach.

The documents in this series describe example implementations of cybersecurity practices that businesses and other organizations may voluntarily adopt. These documents do not describe regulations or mandatory practices, nor do they carry statutory authority.

ABSTRACT

The Transport Layer Security (TLS) protocol is widely deployed to secure network traffic. TLS 1.3 protects the contents of its previous TLS communications even if a TLS-enabled server is compromised. This is known as forward secrecy. The approach used to achieve forward secrecy in TLS 1.3 may interfere with passive decryption techniques that enterprises rely on to have visibility into their TLS 1.2 traffic. Enterprises' authorized network security staff rely on that visibility to protect its data and systems with critical cybersecurity controls to meet operational needs and legal requirements. Adoption of the TLS 1.3 protocol can disrupt current approaches to observing and monitoring internal network communications within an enterprise.

The NCCoE, in collaboration with technology providers and enterprise customers, initiated a project to demonstrate options for maintaining visibility within the TLS 1.3 protocol using several standards-compliant builds that enterprises can use for real-time and post-facto systems monitoring and analytics capabilities.

This publication contains demonstrated proofs of concept along with links to detailed technical information online on NIST pages. This publication also includes links to mappings of TLS 1.3 visibility principles to commonly used security standards and guidelines.

KEYWORDS

bounded lifetime; break and inspect; ephemeral; key management; middlebox; passive decryption; passive inspection; protocol; Transport Layer Security (TLS); visibility.

ACKNOWLEDGMENTS

We are grateful to the following individuals for their generous contributions of expertise and time.

Name	Organization
Ravishankar Chamarajnagar	AppViewX
Michael Ackermann	Blue Cross Blue Shield
Jonathan Chen	F5
Ryan Johnson	F5
Brad Otlin	F5
Kevin Stewart	F5
Nanjaiah Vijayalakshmi	NETSCOUT Corporation
Gina Scinta	Thales Trusted Cyber Technologies
Lauren Brown	JPMorgan Chase & Company

The Technology Partners/Collaborators who participated in this build submitted their capabilities in response to a notice in the Federal Register. Respondents with relevant capabilities or product components were invited to sign a Cooperative Research and Development Agreement (CRADA) with NIST, allowing them to participate in a consortium to build this example solution. We worked with:

Technology Partner/Collaborator	Build Involvement
AppViewX	NETSCOUT Corporation
DigiCert	Not for Radio LLC
F5	Thales Trusted Cyber Technologies
JPMorgan Chase & Company	U.S. Bank Corporation

Technology Partner/Collaborator	Build Involvement
Mira Security, Inc.	

DOCUMENT CONVENTIONS

The terms “shall” and “shall not” indicate requirements to be followed strictly to conform to the publication and from which no deviation is permitted. The terms “should” and “should not” indicate that among several possibilities, one is recommended as particularly suitable without mentioning or excluding others, or that a certain course of action is preferred but not necessarily required, or that (in the negative form) a certain possibility or course of action is discouraged but not prohibited. The terms “may” and “need not” indicate a course of action permissible within the limits of the publication. The terms “can” and “cannot” indicate a possibility and capability, whether material, physical, or causal.

NOTICE: The Information Technology Laboratory (ITL) has requested that holders of patent claims whose use may be required for compliance with the guidance or requirements of this publication disclose such patent claims to ITL. However, holders of patents are not obligated to respond to ITL calls for patents and ITL has not undertaken a patent search in order to identify which, if any, patents may apply to this publication.

As of the date of publication and following call(s) for the identification of patent claims whose use may be required for compliance with the guidance or requirements of this publication, no such patent claims have been identified to ITL.

No representation is made or implied by ITL that licenses are not required to avoid patent infringement in the use of this publication.

Table of Contents

List of Figures

List of Tables

1 Executive Summary

There are sector specific requirements that call for organizations to monitor network activity, protect sensitive data, and demonstrate security controls. Enterprises leverage network traffic visibility in their security operations centers to prevent, detect, and respond to cybersecurity threats. Enterprises moving to newer network security protocol standards such as TLS 1.3 will face challenges for maintaining network traffic visibility.

Modern protocol designers have changed protocols to strengthen security properties that protect the secrecy of historical network traffic. This is possible even if the servers' long-term secret keys are compromised—a property known as *forward secrecy*. However, forward secrecy has created significant challenges for the network visibility strategies used by enterprises.

The National Cybersecurity Center of Excellence (NCCoE), in collaboration with technology providers and enterprise customers, initiated a project demonstrating options for maintaining visibility within an enterprise adopting these new security protocols. The demonstrations are suitable for voluntary adoption across a wide range of enterprise architectures. They are scalable, actionable, and application protocol-agnostic, as well as usable in real-time following post-packet capture.

Enterprises using the Transport Layer Security (TLS) 1.2 protocol without forward secrecy deploy tools and architectural solutions that provide visibility into enterprise traffic within their network. Enterprises have regulatory and compliance requirements to maintain visibility into received network traffic to enable the organization's security monitoring, analysis, and management policies. An enterprise will not be able to use its deployed tools and architectural solutions that provide visibility into enterprise TLS 1.2 traffic to have visibility into TLS 1.3 traffic. This publication includes demonstrated approaches for enterprises to adopt TLS 1.3 to allow enterprises to benefit from the security functionality of TLS 1.3 while maintaining the visibility into received network traffic.

This publication describes the motivation, approach, architecture, build implementation, demonstration scenarios, results, and risk and compliance management characteristics for the demonstrated proofs of concept. The top-level overview provides links to technical details that are contained in online NIST pages. The linked files provide detailed technical information for each demonstration that TLS visibility implementers can adopt in their own environments. The demonstrations in this publication to maintain visibility are not intended as a recommended default or even for common use but to assist in those areas where, as OMB M 22-09 states: "…as agencies segment their networks, move away from intranets, and permit access to enterprise services from any network, inspecting traffic in these environments will become less practical and less valuable over time. In other places, deep traffic inspection may be more valuable and can create less of an increase in attack surface."

2 Introduction

Enterprise cybersecurity depends on identification, protection, detection, and response, as well as recovery policies, mechanisms, and processes. Cryptography—a critical component of cybersecurity—is an important mechanism for protecting enterprise information and processes. Network and system monitoring and analysis of encrypted traffic and the underlying plaintext is often necessary for detecting cyber-attacks and anomalous behavior, understanding their nature, and responding to and recovering from an incident. *Transport Layer Security (TLS)* is a cryptographic protocol that is widely deployed to secure internal enterprise traffic within traditional office networks and enterprise data centers, and connections across the public internet.

The Transport Layer Security (TLS) Protocol Version 1.3 (RFC 8446 [1]) always provides *forward secrecy*. In the legacy TLS 1.2 [2] implementations, forward secrecy is optional, but in TLS 1.3, every session is compatible with forward secrecy, and it is achieved when compatible key management practices are employed, such as deleting keys when they are no longer needed.

Many enterprises have regulatory requirements for visibility into network traffic and stored data. The approach used in TLS 1.3 to achieve forward secrecy conflicts with the passive decryption techniques that are widely used by enterprises to gain visibility into their internal enterprise TLS-protected traffic. The use of forward secrecy conflicts with the passive decryption techniques used today for visibility, resulting in enterprises choosing between using the TLS 1.2 protocol without forward secrecy or adopting TLS 1.3 together with some alternative method for achieving visibility into internal traffic. If an enterprise opts for TLS 1.2, it misses out on the performance enhancements in TLS 1.3 and faces additional risks by relying on increasingly out-of-date protocol implementations. Further, TLS 1.2 will not be updated to support post-quantum cryptographic (PQC) algorithms.

Loss of visibility into received network traffic can impair critical functions such as network and application performance monitoring, security logging, and diagnostics, which undermines the effectiveness of network and security operations and engineering teams. Without the ability to decrypt network data for deep packet inspection (DPI), monitoring, or diagnostics, enterprises will rely on endpoint devices for performance and security information. The incoming network data stream can contain information or perspectives that individual endpoint devices, like workstations, servers, etc., that support a security client are not capable of providing. Reliance on endpoints in the absence of visibility into network traffic introduces security and operational risks for network and data center operations in the following ways:

- Visibility into network data is even more critical when the endpoints are having problems or are in any way compromised. A degraded or compromised endpoint device may fail to report incidents that can be detected within the incoming network data stream.
  - Visibility into network data is essential to enable the ability to discover issues that involve multiple platforms or multiple organizations within the enterprise.
  - Visibility into network data is required when sessions span across devices that do not log information well or at all. Even with devices that do logging well, it is frequently necessary to augment that logging by having visibility to related network data. Even where endpoint data is adequate, it still needs to be collected, consolidated, centralized, and correlated.

Where sessions span domains of control, network data is the only common point at which multiple operators can establish common ground for monitoring, security, and diagnostics.

- When logging of network traffic is turned off or reduced, which is often the case, visibility into network data may be the only alternative to see anomalies. Visibility into network data is preferable when the endpoint (or a middlebox, a network device that often intercepts and manipulates network traffic for security, optimization, or content filtering) platform is incapable of adequate logging without causing utilization or performance issues on the platform.
- There are many security threats that are more easily identifiable or only visible with visibility into network data. Where and how the network data is identified and collected can reveal key information about security threats.
- Staff can analyze network data to ensure that endpoint security agents are operating properly. If an endpoint is compromised or its security agent is not running properly, network data is the only line of defense and the most critical tool for performing related triage and forensics to quickly resolve related issues.
- If network data cannot be decrypted, a security breach, malware, or other compromise can more easily spread throughout the entire organization via a single platform.
- Nearly all attacks occur over the network—leaving behind traces or tracks. To understand questionable traffic, it is necessary to understand the source of the traffic. Without decryption, you cannot gain such insights, nor can you detect, trace, or eliminate malicious actions.
- Root cause analysis is critical to most large organizations. Forensics performed after a breach or other security exposure event depends heavily on DPI and network data to figure out what happened, why, and what can be done about it. Alternatives to DPI require significant time and effort and are generally prohibitively disruptive and expensive. Alternatives to DPI monitoring and analysis of decrypted incoming network data streams include:
    - Re-architecting the enterprise network**:** This is difficult, expensive, and time-consuming, and, even where feasible, is not a short-term solution.
    - Depending on endpoints for management and logging**:** Even if the available endpoint solutions selected are stable, capable, and effective, and are consistent and reliable recorders of all events related to incidents (enhanced logging), this would require building a separate infrastructure capable of collecting, storing, and parsing terabytes (or more) of data. None of this is a simple proposition, and such an infrastructure would require both DPI for certain data and significantly enhanced infrastructure management. Furthermore, if the true root cause is occurring at a middlebox device, endpoints will not see essential information at all.
    - Use of intermediate proxies between application tiers: This approach would add cost, latency, and potential points of failure. The more tiers that a given application has makes proxies less viable and more expensive than enabling visibility of network traffic, as described in this publication. The cost and complexity increases could be enormous in many cases. There may also be situations where intermediate proxies are not possible, such as secure subnets and virtual environments.

A significant constraint in meeting the visibility challenges attendant on TLS 1.3 is the lack of workable approaches that don't change the current TLS 1.3 standard or require the development or adoption of additional or alternative standards.

Our goal is to demonstrate a standards-based reference design and provide users with the information they need to replicate tested TLS 1.3 implementations that permit visibility into network traffic. The demonstration allows an enterprise to benefit from the performance and capability benefits that TLS 1.3 deployment offers and gain visibility into network traffic.

This project addresses known technical and management challenges:

- Secure management of servers' cryptographic keys
- Management of recorded traffic
- Managing expectations of privacy

2.1 Audience

Readers of this publication are assumed to have knowledge of Transport Layer Security and its uses as a cryptographic security protocol that provides communication security over a network. Readers should also be aware that TLS is a security protocol is standardized by the Internet Engineering Task Force (IETF).

2.2 How to use this Guide

This NIST Cybersecurity Practice Guide demonstrates a standards-based reference design and provides users with the information they need to replicate tested TLS 1.3 implementations that permit visibility described in this document.

Technology, security, and privacy program managers and implementers might use this guide to learn about two architectures that will help them maintain visibility into the data transiting their networks.

3 Project Overview

3.1 Background

The NCCoE hosted an industry roundtable in 2018 to assess the scope of the visibility challenges faced by enterprises. NCCoE staff participated in the Center for Cybersecurity Policy's 2019 workshop [3] on enterprise visibility and helped identify a set of baseline criteria for the acceptability of solutions for visibility challenges. The NCCoE adopted the criteria without change and hosted a virtual workshop focused on TLS 1.3 in September 2020 [4] where participants identified the following options for maintaining visibility:

- Endpoint mechanisms that establish visibility, such as enhanced logging
- Network architectures that inherently provide visibility, e.g., using overlays or incorporating middleboxes
- Key management mechanisms that defer forward secrecy to maintain current levels of network visibility

- Innovative tools that analyze network traffic without decryption
- Deploying alternative standards-based network security protocols where forward secrecy is optional or unsupported.

In May 2021, the NCCoE published a project description for a TLS 1.3 visibility project that featured use case scenarios for the implementation of potential solutions, as discussed in the workshops. Collaborators participating in this project submitted their capabilities in response to NIST's open call in the *Federal Register* for all sources of relevant security capabilities from academia and industry (vendors and integrators). Those respondents with relevant capabilities or product components were selected as collaborators and signed a Cooperative Research and Development Agreement (CRADA) to participate with NIST in a consortium to build the example solution and to develop the architectures and demonstration scenarios featured in this practice guide.

3.2 Solution

The project demonstrates approaches and practices that meet common compliance, operations, and security requirements while gaining the performance and capability benefits that TLS 1.3 deployment offers.

We focus on enterprise data center environments; this is the environment most impacted by the changes to TLS 1.3 regarding visibility. This includes on-premises data centers and hybrid cloud deployments hosted by a third-party data center or public cloud provider. We use real-world visibility approaches that utilize current or emerging components, proprietary vendor products, and commercially viable open-source solutions. We include approaches that restore visibility into encrypted data in transit, such as alternative key establishment mechanisms and how to manage NIST's and industry's standards. The following are out of scope and not impacted by our proposed solutions:

- Information transmitted over the public internet (e.g., connections between enterprise users and services on the public internet).
- Emerging deployment models leveraging encrypted transport to protect protocols that were previously in the clear, such as DoT (Domain Name System [DNS] over TLS) [5], DoH (DNS over Hypertext Transfer Protocol Secure [HTTPS]) [6], and DoQ (DNS over QUIC) [7]. DoT, DoH, and DoQ may be the subject of future NCCoE work.

This project does not change or replace the RFC 8446 [1] standard. Our proposed solutions provide secure management of servers' cryptographic keys, securely manage recorded traffic, and consider privacy via a broad set of options, including:

- **Key-management mechanisms** that defer forward secrecy until all copies of keying material needed to maintain current levels of network visibility are deleted, such as copies retained to support passive decryption and inspection. ***Passive decryption and inspection*** is referred to throughout this publication as ***passive inspection***. Passive inspection involves inspecting encrypted traffic without disrupting the data flow or requiring any changes to the network or applications using TLS.
- **Network architectures** that provide visibility, such as using overlays or incorporating middleboxes (see RFC 3234: Middleboxes: Taxonomy and Issues [8]).

The TLS 1.3 visibility project focuses on passive inspection and middlebox solutions to avoid losing visibility characteristics and to avoid vulnerabilities identified from continuing to use TLS 1.2.

To achieve visibility through key management (via passive inspection), we demonstrate two technical mechanisms for servers whose traffic is of interest to the enterprise, as follows:

- The enterprise provisions bounded-lifetime Diffie-Hellman (DH) key pairs for TLS 1.3 servers that are used in ephemeral key exchanges. This approach includes a purely static deployment that can also use key pairs for a short period of time.
- The enterprise collects and retains the per-session symmetric session keys used to encrypt the connections instead of provisioning DH key pairs.
  - Some aspects of analytics functions need enterprise visibility into their encrypted traffic. This may require combining network architecture and key-management techniques to achieve operational visibility. Therefore, this project demonstrated an architecture that achieves visibility inside the data center using tools that break and inspect traffic. These middleboxes are commonly used at the enterprise edge to achieve real-time visibility. We also demonstrated how an enterprise can access historical data through key management-based solutions.
  - With passive inspection solutions, a key distribution function manages DH keys and symmetric traffic keys. Both are retained by a key distribution function while corresponding encrypted traffic is either decrypted, destroyed, or no longer available.
  - Authorized systems that examine traffic obtain the appropriate keys from the key distribution function. The solution incorporates components to retain traffic for retrospective applications, e.g., troubleshooting and cybersecurity forensics. Stored traffic is retained in encrypted form until policy conditions (e.g., retention time limits) are met. The data is then deleted by the storage function. The resulting solutions protect keys and data against misuse or compromise, and they don't leave recorded traffic at risk of compromise indefinitely. The solutions include mechanisms and procedures used to transmit, store, provide access to, and use cryptographic keys that can perform comprehensive deletion of decryption keys when defined temporal or data protection limits are met.

Since TLS 1.3 is designed to allow client/server communication in a way that prevents eavesdropping, the solutions also assume out-of-band notification of the visibility policy.

4 Architecture and Builds

The project's technology collaborators have offered products and insights to help organizations gain more visibility into traffic protected by the improved TLS 1.3 protocol. This section identifies the project collaborators, components of the functional architecture employed, and the collaborators' products we used to implement the functional architecture.

4.1 Project Collaborators

The following organizations have collaborated with the NCCoE to demonstrate how to maintain real-time and post-facto visibility into enterprise network traffic when using TLS 1.3. Real-time visibility allows for threat detection during data exchange, while post-facto visibility enables analysis after the

fact, such as forensics analysis, to understand anomalies and respond to or recover from security incidents.

4.1.1 AppViewX

AppViewX is an automated certificate lifecycle management (CLM) solution that simplifies public key infrastructure (PKI) and certificate management. It combines automation, security, and insights to meet all enterprise PKI and key management needs. AppViewX CERT+ features are purpose-built to address operational and security challenges of certificate and key management to help organizations prevent application outages and security breaches. AppViewX's capabilities include discovering all certificates across complex enterprise environments, building and maintaining central inventories, provisioning private and public trust certificates from any certificate authority (CA), expiring certificate alerts, and fully automated renewals and revocations.

AppViewX partnered with NETSCOUT to contribute a prototype TLS 1.3 key governance platform that it plans to formalize as an open industry standard. Key governance platform pairs with a Secure Key Orchestration initiative to secure and automate the management of all encryption keys across distributed and hybrid enterprise environments. The AppViewX Cloud-native Identity and Security Platform is used in critical infrastructures to reduce cybersecurity risk and meet security compliance requirements. Thanks to streamlined automation workflows, the AppViewX Platform supports enterprise-wide central certificates and key governance and lifecycle management. The modular AppViewX Platform and its CERT+ and PKI+ products are delivered as a service to address digital and machine identity challenges. AppViewX provisions private and public trust certificates from any CA, alerts to expiring certificates, and automates renewals and revocations. CERT+ is an automated certificate lifecycle management (CLM) solution for simplified PKI and certificate management. PKI+ is a turnkey PKI-as-a-Service for private trust use cases that reduces PKI hardware requirements, simplifies private PKI architectures, and sets up tailored custom CAs. For more details, visit https://www.appviewx.com.

4.1.2 DigiCert

DigiCert provides scalable TLS and PKI solutions for identity and encryption. The company is known for its expertise in identity and encryption for web servers and Internet of Things devices. DigiCert supports TLS/ Secure Sockets Layer (SSL) and other digital certificates for PKI deployments at any scale through its certificate lifecycle management platform, CertCentral®. The company provides enterprise-grade certificate management platforms, responsive customer support, and advanced security solutions.

DigiCert's CertCentral web-based platform allows provisioning and managing publicly trusted X.509 certificates for TLS and code signing and a variety of other purposes. After establishing an account, clients can log in, request, renew, and revoke certificates via a browser. Multiple roles can be assigned within an account, and a discovery tool can inventory all certificates within the enterprise. In addition to certificate-specific features, the platform offers baseline enterprise SaaS capabilities, including role-based access control (RBAC), Security Assertion Markup Language (SAML), single sign-on (SSO), and security policy management and enforcement. All account features are fully compatible with the web portal and a publicly available API. Learn more about DigiCert at https://www.digicert.com.

4.1.3 F5

F5, Inc. is a publicly-held American technology company specializing in application security, multi-cloud management, online fraud prevention, application delivery networking, application availability & performance, network security, and access & authorization. F5 is headquartered in Seattle, Washington, with an additional 75 offices in 43 countries, focusing on account management, global services support, product development, manufacturing, and software engineering. F5 offers application delivery controller technology, application layer automation, multi-cloud, and security services. The company offers modules on its proprietary operating system, TMOS (Traffic Management Operating System), including Local Traffic Manager, Advanced Web Application Firewall, Domain Name Service, and Access Policy Manager. The modules offer the ability to deploy load balancing, Layer 7 application firewalls, SSO (for Active Directory [AD]), Azure AD, Lightweight Directory Access Protocol (LDAP), and enterprise-level virtual private networks. F5's BIG-IP is available as a hardware product and a virtual machine (BIG-IP Virtual Edition) that is cloud-agnostic and can be deployed on-premises in a public and/or hybrid cloud environment.

F5 has contributed the BIG-IP SSL Orchestrator to the TLS 1.3 visibility project, which provides security solutions with enhanced visibility into encrypted traffic through dynamic service chaining and policy-based traffic steering. Purpose-built for TLS decryption, the SSL Orchestrator applies context-based intelligence to direct encrypted traffic across the security stack, ensuring optimal tool availability and performance. It centralizes TLS decryption for multiple security tools, simplifies management within complex architectures, and supports next-generation encryption protocols—allowing organizations to efficiently scale and adapt their security infrastructure. The BIG-IP SSL Orchestrator inspects encrypted traffic by decrypting, routing it through security controls, and re-encrypting it. This enables the discovery of hidden threats and multi-stage attack prevention. Designed to integrate flexibly with existing architectures, the SSL Orchestrator supports security stack orchestration—providing flexible deployment options that allow enterprises to optimize visibility and defend against evolving threats across their network. Learn more about F5 at https://www.f5.com/.

4.1.4 JPMorgan Chase & Co.

JPMorgan Chase & Co. is an American multinational financial services firm headquartered in New York City and incorporated in Delaware. It is the largest bank in the United States and the world's largest bank by market capitalization. JPMorgan Chase manages large-scale network operations with many customers and partners. The network traffic is TLS-protected. Security and reliability considerations require continuous monitoring and analytics to support threat and incident detection, auditing, and forensics. The analytics processes require real-time and post-facto visibility into traffic metadata and contents. As such, JPMorgan Chase is providing content, protocol, and performance requirements and constraints information that supports the project's functional objectives. Learn more about JPMorgan Chase at https://www.jpmorganchase.com/.

4.1.5 Mira Security

Mira Security delivers standalone TLS visibility solutions, allowing existing, unmodified enterprise security tools to detect and block threats hidden inside encrypted traffic flows. Mira Security's technology is embedded in solutions from many companies, as well as being available directly from

Mira. Their Encrypted Traffic Orchestrator (ETO) software supports all the latest TLS standards—providing visibility into encrypted traffic without weakening the security profile of the connection. ETO software can be deployed as a physical or virtual appliance or in public cloud environments, delivering consistent features and functionality across deployments.

The ETO offers a transparent TLS visibility solution that decrypts traffic for security tools, enabling threat detection in encrypted flows. ETO integrates seamlessly at the network layer without requiring changes to network architecture and provides fine-grained policy controls for compliance with privacy and security standards. Physical ETO appliances support interface speeds from 1 Gbps to 40 Gbps, with decryption capacity up to 100 Gbps; virtual ETO (vETO) supports decryption up to 5 Gbps on KVM and ESXi, with similar capability in AWS. The optional Category Database service enhances ETO's policy controls by enabling category-based rules, such as excluding decryption of “health care” traffic. ETO can be managed via WebUI or REST API, integrating with existing frameworks. For large deployments, the Mira Central Management System (CMS) centralizes policy management, licensing, and configuration across multiple devices. Learn more at https://mirasecurity.com.

4.1.6 NETSCOUT

NETSCOUT Systems, Inc. (NETSCOUT) protects digital business services against disruptions in availability, performance, and security. NETSCOUT combines its patented smart data technology with smart analytics and provides real-time, pervasive visibility and insights to accelerate and secure customers’ digital transformation. NETSCOUT’s approach aims to transform the way organizations plan, deliver, integrate, test, and deploy services and applications. Its nGenius service assurance solutions provide real-time, contextual analysis of service, network, and application performance. The mission of NETSCOUT is to protect the global industry from the risks of disruption, allowing solutions to network performance and security problems. In support of its mission, NETSCOUT provides software solutions that support service assurance, advanced cyber threat and distributed denial of service (DDoS) protection, and business analytics/big data areas of its customers’ business.

NETSCOUT’s Visibility Without Borders Platform contributes to the TLS 1.3 visibility project with its nGeniusONE Service Assurance platform, vSTREAM™ virtual appliance, Omnis Cyber Intelligence console, and CyberStream network security sensors. The nGeniusONE platform offers comprehensive performance monitoring and troubleshooting for IP-based services, integrating real-time monitoring, historical analysis, and multi-layered analytics for holistic service management. The vSTREAM™ virtual appliance extends Adaptive Session Intelligence™ (ASI)-based visibility to virtual and cloud environments, supporting traffic monitoring within hosts or as an aggregation point across multiple hosts. Seamlessly integrated with nGeniusONE, nGeniusPULSE, and NETSCOUT Smart Edge Monitoring, it supports consistent service-critical visibility across infrastructures.

Omnis Cyber Intelligence acts as the central console for the Omnis Security platform, analyzing data from CyberStreams, ISNGs, and vSTREAMs to detect cyber threats, enriched by ATLAS and third-party intelligence feeds. Alerts and data can be exported to third-party SIEMs and data lakes for extended analysis. Omnis CyberStream uses threat detection and machine learning to detect known and zero-day threats. Its Network Detection and Response (NDR) platform integrates with SIEM/SOAR and XDR systems, providing a unified interface for efficient security management and rapid response. CyberStream sensors deploy in any environment, converting packet data into detailed Layer 2-7

metadata for comprehensive network visibility and threat detection. Learn more about NETSCOUT at https://www.netscout.com/.

4.1.7 Not for Radio

Since 2013, **Not for Radio** (NFR) has provided solutions to complex challenges in communication networks for both corporate and government customers, with deployments in internet and telecommunication infrastructure as well as high-performance computing fabrics. NFR's contribution to the project is its Encryption Visibility Architecture™ (EVA™) product, which offers a flexible software solution for maintaining data visibility in enterprise networks following the deployment of TLS 1.3 while supporting additional protocols such as legacy TLS and IPsec.

EVA is designed to be minimally intrusive with respect to the diversity of existing security postures, compliance regimes, performance requirements, and orchestration technologies typically found in service operator environments. The demonstration systems constructed for this project employ NFR's Encryption Visibility Agent™ (EVA™) in its Bounded Lifetime Key Control mode, with an external key management system configured as the source of the bounded-lifetime key material. With this configuration, the Agent runs within the applications of interest and enforces the use of the controlled, bounded-lifetime Diffie-Hellman key material in TLS 1.3 sessions. Importantly, the Agent's operation does not introduce new pathways for the lateral movement of malware by requiring the relaxation of any platform security mechanisms. Other modes of operation of the EVA, such as high-performance and fully deterministic reporting of per-session key material, as well as distributed bounded-lifetime key generation, are not used in this demonstration. Additional components of the Encryption Visibility Architecture™ family, designed to address scalability and integration challenges within larger deployments, are likewise not used.

4.1.8 Thales Trusted Cyber Technologies

Thales Trusted Cyber Technologies is a U.S. provider of cybersecurity solutions dedicated to the U.S. Government. It protects data from the core to the cloud to the edge with a unified approach to data protection. Thales' solutions reduce the risks associated with critical attack vectors and address stringent encryption, key management, and access control requirements. In addition to the core solutions developed and manufactured in the U.S. specifically for the Federal Government, Thales sells and supports third-party, commercial-off-the-shelf solutions. To mitigate the risks associated with procuring data security solutions developed outside of the U.S, Thales operates under a Proxy Agreement with the Defense Counterintelligence and Security Agency (DCSA) for Foreign Ownership, Control, or Influence (FOCI) and Committee on Foreign Investments in the United States (CFIUS) National Security Agreement.

For this project, Thales contributed its hardware security module (HSM), a dedicated cryptographic processor that is specifically designed for the protection of the crypto key lifecycle. The HSM acts as a trust anchor that protects the cryptographic infrastructures by securely managing, processing, and storing cryptographic keys inside a hardened, tamper-resistant device. Thales HSMs always store cryptographic keys in hardware. They provide a secure crypto foundation, as the keys never leave the intrusion-resistant, tamper-evident, FIPS 140-validated [9] appliance. Since all cryptographic operations occur within the HSM, strong access controls prevent unauthorized users from accessing sensitive

cryptographic material. Thales also implements operations that make the deployment of secure HSMs as easy as possible. They are integrated with the Thales Crypto Command Center for quick and easy crypto resource partitioning, reporting, and monitoring. Learn more about Thales Trusted Cyber Technologies at https://www.thalestct.com/.

4.2 Architecture and Builds

Some aspects of the analytics functions requiring enterprise visibility into its encrypted TLS 1.3 traffic may consider combining network architecture and key-management techniques to achieve operational visibility. These functions may include:

- identifying the causes of network performance degradation or failures
- key management-based communications failures
- detection and identification of anomalous received data
- identification of sources of anomalous data
- detection of encrypted traffic from unauthorized sources
- extraction of enterprise data to anomalous destinations.

This project aims to develop and test an architecture that provides visibility within an enterprise data center. This is achieved using tools that intercept and decrypt traffic without altering the traffic flow between the TLS clients and servers, without changing the TLS 1.3 protocol. In this demonstration project, we examine TLS 1.3 deployment within the enterprise data center and address mechanisms that can support access to historical data by leveraging key management-based and middlebox solutions.

This NIST Cybersecurity Practice Guide addresses the challenge of maintaining visibility into network traffic encrypted with TLS 1.3 within enterprise data centers. It focuses on securely managing servers' cryptographic keys, recorded traffic, and privacy expectations. Our builds demonstrate real-time decryption, analysis, or post-facto decryption and TLS 1.3 encrypted traffic described by one of the following:

- Bounded-lifetime DH keys on the TLS server
- Export of TLS session keys from the TLS server
- Break and inspection of TLS traffic using a middlebox
- Open Systems Interconnection (OSI) Data Link Layer 2 cryptography
- OSI Network Layer 3 cryptography

4.2.1 System Architecture Functions

Below are the components that comprise the TLS 1.3 visibility architecture.

- **Server Components**: Handle services like HTTPS and email, manage network resources, generate session keys, negotiate encryption protocols, and integrate with key management infrastructure.
- **Client Components**: Initiate encrypted traffic for users, devices, and processes that interact with servers to request certificates and keys. They are typically located outside of data centers.
- **Network Tap Function**: Copies network traffic for logging and monitoring, aiding in the detection of malicious activity or security threats.

- **Break and Inspect Middlebox**: Decrypts, inspects, and re-encrypts traffic to identify threats before the data is transmitted into or out of the network.
- **Real-Time Decryption**: Decrypts and forwards traffic with minimal delay to support real-time security needs.
- **Real-Time Analytics**: Processes data quickly for immediate threat detection and response, providing insights on network performance and potential anomalies.
- **Post-Facto Decryption and Analytics**: Decrypts and stores encrypted data for later analysis, such as forensic investigations, ensuring secure handling and disposal of data.
- **Key Management Agent**: Provides a secure interface for provisioning TLS server keys and implementing policies for key activation and expiration.
    - **Key Capture and Registration Agent:** Captures the session keys at the time they are generated and registers the session keys with the Key Governance Platform.
- **Enterprise PKI**: Manages digital certificates and validates identities, binding them to cryptographic keys for secure communication. The trust anchors for an Enterprise PKI are not expected to be known outside the enterprise.
- **Key Governance**: Oversees the lifecycle of certificates and keys, including issuance, renewal, revocation, and secure storage.
- **Key Source**: A secure, FIPS 140-validated component that generates cryptographic keys for use within the TLS 1.3 project.

4.2.2 High-Level Passive Inspection Architecture Overview

The figures below depict the functional components of a passive decrypt and inspect demonstration architecture. Figure 4-1 depicts passive inspection using rotated bounded-lifetime DH keys on the destination TLS server. This approach can be used to capture decrypted traffic for real-time analysis, incoming traffic for post-facto or historical analysis, or both. Note that the clients internal to the enterprise receiving TLS 1.3-protected traffic from the TLS server are not depicted in Figure 4-1.

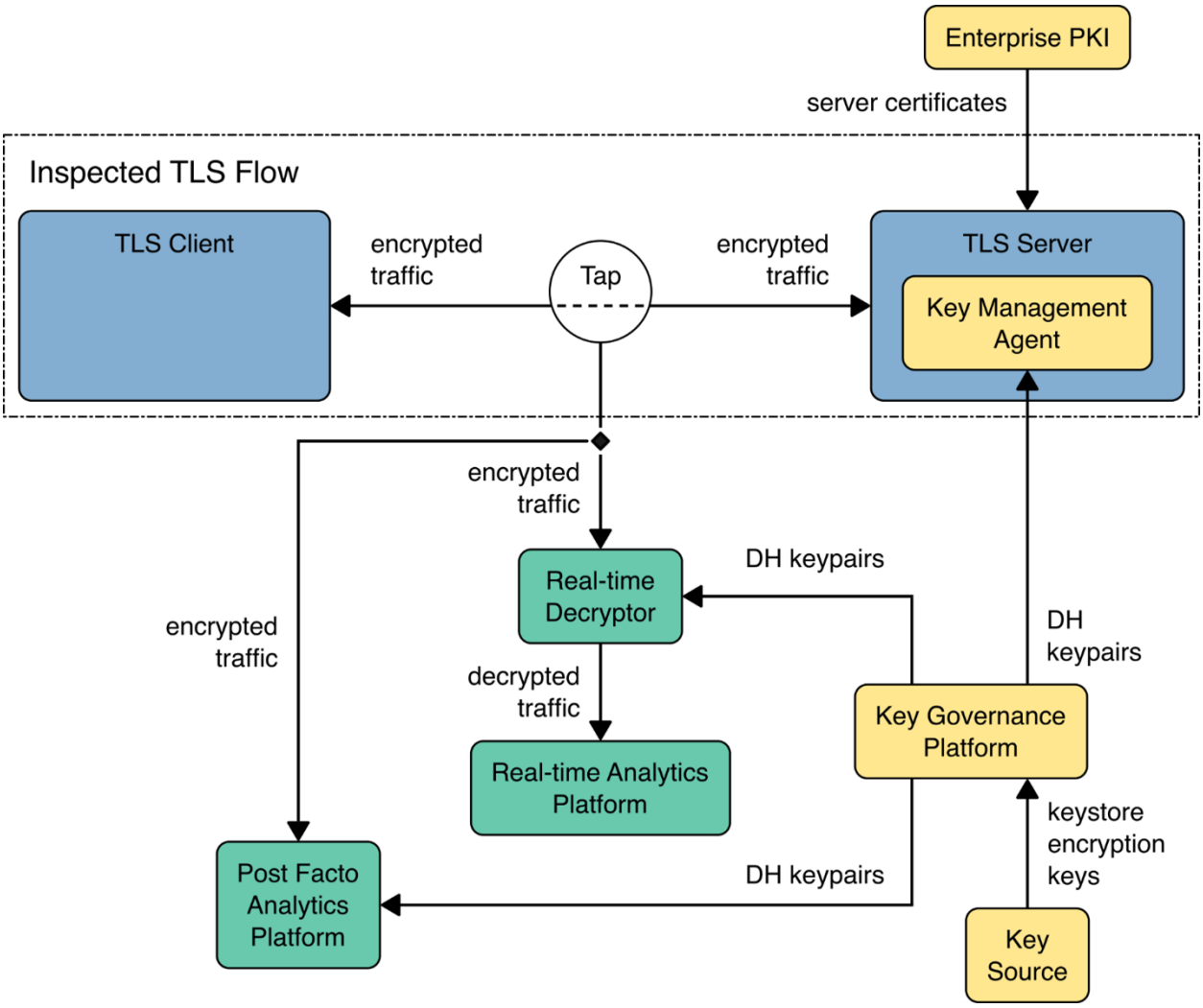

Figure 4-1 Middlebox (Break and Inspect) Functional Architecture

Figure 4-2 depicts passive decryption and inspection using exported session keys. The architecture permits real-time analysis of decrypted TLS traffic and post-facto analysis of stored encrypted traffic. Note: Exported session keys can be used to decrypt TLS traffic irrespective of the session's TLS version and cipher suite. In addition to exporting the session keys, the Key Management Agent also exports the Client Random from the TLS handshake to allow real-time and post-facto decryption devices to match session keys to network flows.

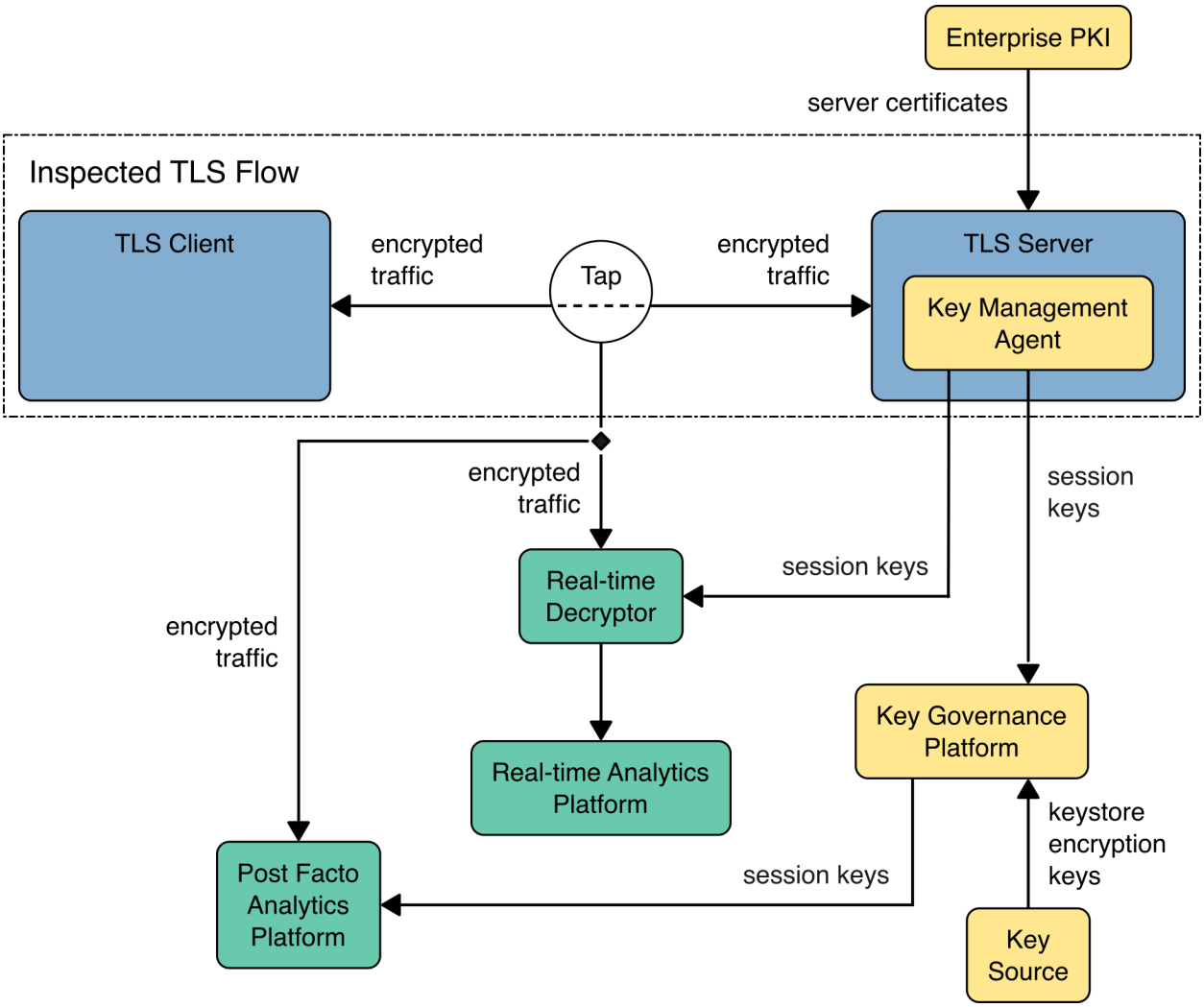

Figure 4-2 Passive Inspection - Exported Session Key Functional Architecture

The function of each component used for passive inspection is described as follows:

- **TLS Client Devices:** Devices that initiate encrypted traffic.
- **Network Tap:** Component that provides a copy of traffic from a network segment.
- **Real-Time Decryption:** Passive decryption component that decrypts and forwards the copied traffic.
- **Real-Time Analytics Platform:** Set of tools for examining decrypted payloads to identify undesired characteristics.
- **Traffic Capture Platform:** Encrypted storage of captured traffic to allow subsequent analytics of captured traffic. This can be encrypted storage of captured decrypted traffic or storage of the captured original encrypted traffic.
- **Key Governance Platform:** Security module performing storage and distribution of keys (e.g., discover, create, renew, provision, revoke, and destroy certificates and keys).

Bounded-lifetime DH keys are pushed to the TLS server and passive decryption device to provide real-time decryption. They are also stored for future use by decryption solutions that work with captured encrypted sessions. Exported session keys and flow identification data are obtained from the Session Key Capture agent or the decryption platform. These keys enable real-time decryption and are stored for future use by decryption solutions that handle captured encrypted sessions.

- **TLS Server:** Peer for encrypted traffic that generates session keys, negotiates encryption protocols, and connects to key management infrastructure.
    - Key Management Agent
        - **Bounded-Lifetime DH Key Management Agent (Figure 4-1):** Receives the bounded-lifetime keys from the key governance platform and enables their use by the TLS server according to key governance platform policy.
        - **Key Capture and Registration Agent (Figure 4-2):** Captures the session keys at the time they are generated and registers the session keys with the Key Governance Platform and the passive real-time decryptor.
- **Enterprise Public Key Infrastructure (PKI):** CA that provides enterprise public key certificates.

Note: Information transfers within an enterprise and any information stored on/by the analytics platform require cryptographic protection or compensating physical controls.

4.2.3 High-Level Middlebox Architecture Overview

To achieve necessary visibility into encrypted TLS 1.3 traffic, some enterprise analytics functions may require a middlebox architecture that integrates network and key-management techniques. These functions may include identifying network performance issues, managing key-based communication failures, and detecting anomalous data sources or unauthorized encrypted traffic. The project scope includes a demonstration of middleboxes within the data center that "break and inspect" traffic for real-time analysis, commonly deployed at the enterprise edge. Figure 4-3 illustrates the components of the Break and Inspect (B&I) architecture used in this demonstration.

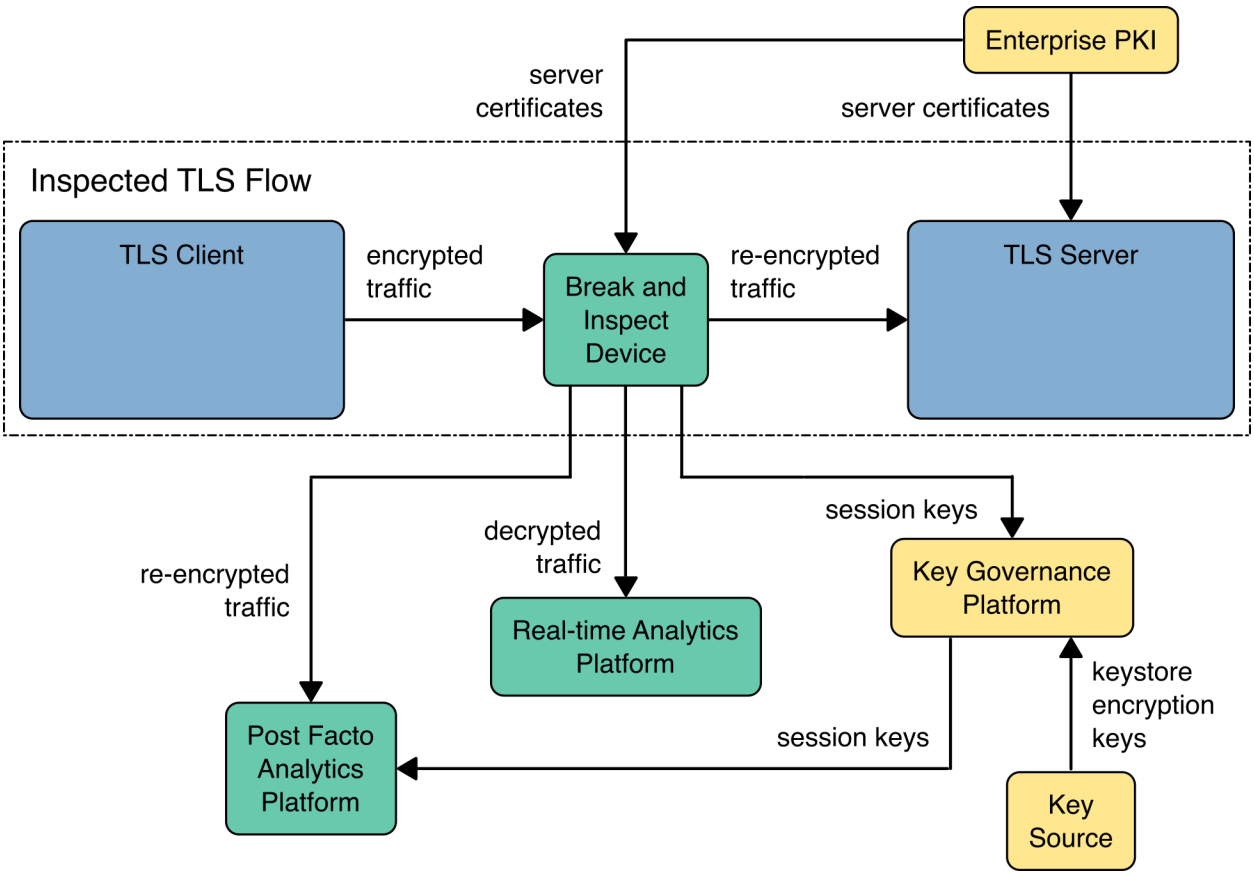

Figure 4-3 Components of Break and Inspect Middlebox Architecture

Below are descriptions of the B&I middlebox components:

- **TLS Client Devices:** These devices initiate encrypted traffic and may reside outside the data center. However, B&I is not using bounded-lifetime DH or ephemeral key reporting to gain visibility. As such, the TLS 1.3 sessions from both an external client to the B&I device and from the B&I device to the server have forward secrecy.
- **Break and Inspect Component:** Component that terminates, decrypts, and re-encrypts/re-initiates TLS traffic.
- **Real-Time Analytics Platform:** Set of tools that examine unencrypted payloads to identify undesired characteristics.
- **Traffic Capture Platform:** Encrypted storage of captured decrypted traffic or storage of the captured original encrypted traffic that enables subsequent analytics of captured traffic.
- **Key Governance Platform:** Security module performing storage and distribution of ephemeral session keys and associated flow identification data provided by the B&I device for later use by a passive decryption device working on captured encrypted traffic.
- **TLS Server:** Peer for encrypted traffic that generates session keys, negotiates encryption protocols, and connects to the enterprise PKI infrastructure.

- **Enterprise PKI:** CA that provides enterprise key certificates.
- Note: Information transfers within the enterprise and any information stored on or by the analytics platform require cryptographic protection or compensating physical controls. Also, in the example above, the B&I device feeds analytics tools with a copy of the decrypted TLS traffic (i.e., the analytic tool is passive and consumes the feed). B&I devices are capable of feeding the decrypted TLS traffic to inline security tools, which may modify the decrypted traffic before returning it to the B&I device for re-encryption and forwarding it to the final destination. In the use case above, with passive analytic tools, the end-to-end payload between client and server is unmodified, whereas the use of inline tools may result in modification.

5 Build Implementation

The builds described in this guide are implementations of the TLS 1.3 visibility reference architectures described in Section 3. The first reference architecture achieves visibility into TLS 1.3 traffic by active management provisioning and using bounded-lifetime DH keys on the TLS server. For real-time visibility, an internal decryptor leverages the known DH key and information in the captured TLS handshake to determine the session key to decrypt the traffic. Similarly, for post-facto visibility, the analytics platform uses the known DH key and information in the captured TLS handshake to identify and determine the session key needed to decrypt the traffic.

The second reference architecture achieves visibility by capturing and exporting the TLS session key negotiated between a TLS client and TLS server to the key governance platform. For real-time visibility, an agent on the TLS server captures the session key as it is generated and immediately registers the session key material. For performance reasons, this build initially sends the key material to the decryptor, which registers the key material in the key governance platform. The decryptor then uses the session key to decrypt the session.

Finally, two reference builds achieve visibility through the breaking and inspection of TLS traffic using a middlebox. One middlebox builds and operates on traffic at the OSI Layer 3 level, and the other operates on the OSI Layer 2. There are two TLS sessions: the first between the TLS client and the middlebox, and the second between the middlebox and the TLS server. The decrypted traffic is copied between the two connections. To support post-facto decryption, the TLS session keys of one or both TLS sessions are registered with the key governance platform

Each architecture contains a subset of 12 components, including TLS 1.3 servers, TLS 1.3 clients, network tap(s), break and inspect middlebox(es), real-time TLS traffic decryption, real-time TLS traffic analysis, post-facto traffic decryption and analysis, key management agents, key capture and registration agents, enterprise PKI, key governance, and key sources. The architecture's daemons generate or capture sensitive key material, and each one ensures that:

- Sensitive key material generated by or captured in the key governance platform is stored in encrypted form with access to the master encryption key managed by a hardware security module.
- All key material is transferred between components only on the dedicated, limited-access network using TLS or SSH connections.

5.1 Passive Inspection Architecture Builds

Passive inspection architectural options support the capture of traffic from monitored network segments, providing mechanisms to decrypt the traffic for immediate analysis or forwarding encrypted traffic for storage structured with session information and the appropriate key information. This approach does not terminate or otherwise modify traffic between the TLS clients and servers.

5.1.1 Bounded-Lifetime Key Pair (Bounded-Lifetime Diffie-Hellman)

For TLS using bounded-lifetime key pairs, the key governance platform provisions the TLS server with bounded-lifetime DH key pairs via a key management agent running on the server. The TLS server then uses the provisioned key pairs instead of performing ephemeral generation during the normal TLS handshake. Due to differences in definitions of "ephemeral," the bounded-lifetime DH key pairs are considered ephemeral keys in RFC 8446 [1] and static keys in SP 800-56A [10]. The key governance platform provisions the server with new bounded-lifetime DH key pairs on a frequent basis via the agent. A decryption platform that has the bounded-lifetime DH key pairs used by the TLS server to establish TLS sessions can decrypt all TLS sessions to the server for the period during which the server uses those DH key pairs.

5.1.1.1 Real-Time (RT) Decryption Flow

Figure 5-1 depicts the elements involved in the bounded-lifetime DH demonstration of real-time decryption capabilities. The descriptive detail for passive inspection using bounded lifetime DH is provided here.

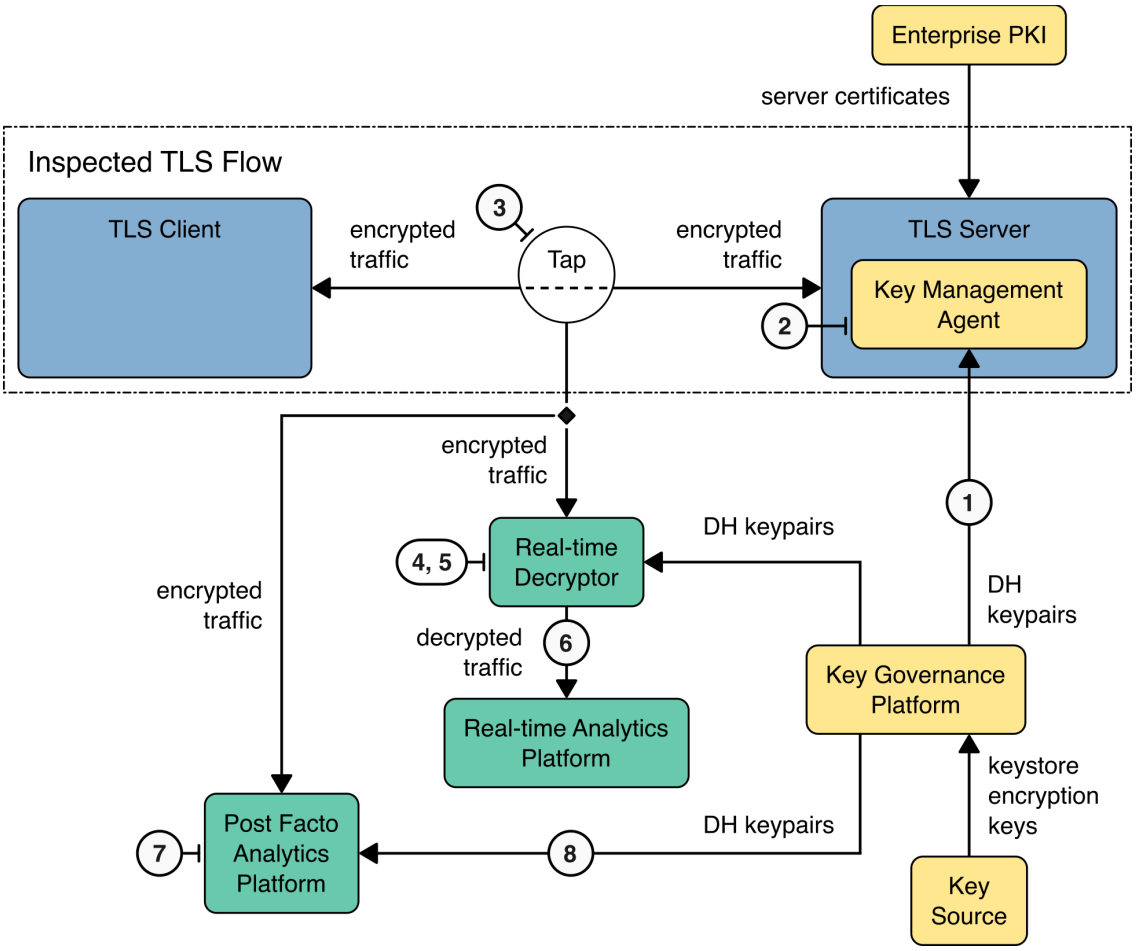

Figure 5-1 Real-Time Bounded-Lifetime DH Passive Inspection Flow

The demonstration of real-time decryption using bounded-lifetime DH keys executed the following sequence:

[1] Before an epoch* begins, the Key Governance Platform generates a new bounded-lifetime DH key pair and pushes it to the TLS Server and the RT Decryptor.
[2] When the epoch begins, the Key Management Agent configures the TLS Server to use the new bounded-lifetime DH key pair to negotiate a new TLS session with the client.
[3] The Network Tap captures encrypted packets between client and server and forwards them to the RT Decryptor.
[4] The RT Decryptor calculates the symmetric session keys from the captured TLS handshake and the known server key pair.
[5] The traffic is decrypted using the calculated session symmetric keys.
[6] The decrypted traffic is forwarded to the RT Analytics Platform.

*An epoch is the rotation period for keys determined by the enterprise key governance platform.

5.1.1.2 Post-Facto Decryption Flow

Figure 5-2 depicts the elements involved in the bounded-lifetime DH demonstration of post-facto decryption capabilities.

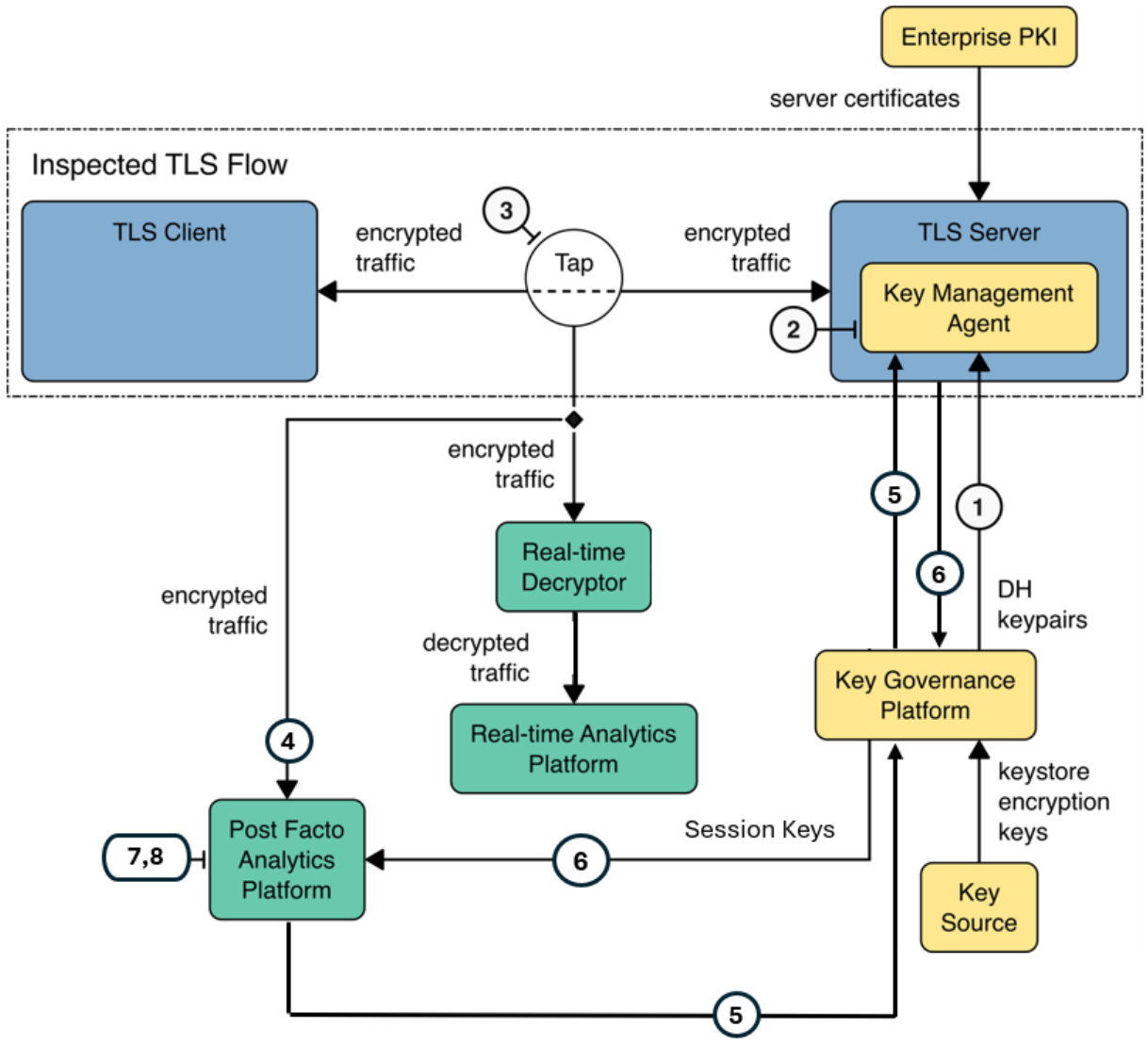

Figure 5-2 Post-Facto Bounded-Lifetime DH Passive Inspection Flow

5.1.1.3 Post-Facto Decryption Flow

The demonstration of storage of traffic for post-facto decryption and analysis using bounded-lifetime DH keys executed the following sequence:

[1] Before an epoch begins, the Key Governance Platform generates a new bounded-lifetime DH key pair and pushes it to the Key Management Agent on the TLS Server (and RT Decryptor).
[2] When the epoch begins, the Key Management Agent configures the TLS Server to use the new bounded-lifetime DH key pair to negotiate a new TLS session with the client.
[3] The Network Tap captures encrypted packets between client and server and forwards them to the Post-Facto Analytics Platform for capture and storage.
[4] The Post-Facto Analytics Platform selects the stored traffic stream to be decrypted.

[5] On a per-traffic stream basis, the Post-Facto Analytics Platform requests the server key pair from the Key Governance Platform using the TLS Server and the epoch of the traffic stream.
[6] The Post-Facto Analytics Platform uses a PKCS#11 exchange to send information from the TLS handshake to the Key Governance Platform, which uses the server key pair to calculate and return the session keys. This means the Analytics Platform doesn't require access to the server keys.
[7] The traffic is decrypted using the calculated symmetric keys.
[8] Decrypted traffic is now available for analysis.

(Note: Key stores are generated by the Key Governance Platform.)

5.1.1.4 Bounded-Lifetime DH Passive Inspection Laboratory Build Components

The software or services used for each of the architecture components in the lab build for this reference architecture are in Table 5-1: .

Table 5-1: Build Components for the Passive Decryption Using Bounded Life-time Server Keys Reference Architecture

Architecture Component	Collaborator	Product Information
Enterprise PKI	DigiCert	• CertCentral Certificate Management Platform
TLS Client		• Mozilla Firefox Browser • Microsoft Edge Browser
Tap		• VMWare vSphere Distributed vSwitch port mirroring
TLS Servers		• NGINX reverse proxy and containerized Keycloak IdAM application • NGINX reverse proxy and containerized Python application • MariaDB Database Server • Postfix SMTP Server and Dovecot IMAP Server
Key Management Agent	NotForRadio	• Encryption Visibility Architecture Encryption Visibility Agent (EVA) Version 0.7.0
Real-time Decryptor	MIRA	• Encrypted Traffic Orchestration (ETO) Version 3.1.0-2024.11.28-4781 Virtual Appliance
Key Governance Platform	AppViewX	• AppviewX Version 2020.3.10 build 55 Virtual Appliance • AppviewX Software Security Module Version 2.6.1 Virtual Appliance
Real-time Analytics Platform		• nGeniusONE Version 6.3.5 build 3184

	NetScout	• Omnis Cyber Intelligence (OCI) Version 6.3.5 build 3184
Post-Facto Analytics Platform		• vSTREAM ISNG Version 6.3.5 build 3184 • vCYBERSTREAM ISNG Version 6.3.5 build 3184
Key Source	Thales TCT	• Luna Network HSM T-5000 Firmware: Version 7.11.1 FIPS-validated

5.1.1.5 Installation and Configuration for Bounded Lifetime Diffie-Hellman

Instructions for installation and configuration of bounded lifetime DH builds are here.

5.1.2 Decryption Using Exported Session Keys

In the exported session key approach, decryption is achieved by either obtaining TLS session keys from the key governance platform or receiving them from the Session Key Capture agent for real-time decryption. This architecture uses agents operating on the TLS servers to capture the ephemeral session keys at the time they are negotiated for a TLS flow. In the case of capture of encrypted flows for post-facto or historical analysis, these agents register the captured keys with the key governance platform. These keys can be retrieved from the key governance platform using the TLS session's client-random-id as the flow identification mechanism. This decryption mechanism works regardless of the TLS version and cipher suite negotiated between the client and server. **Error! Reference source not found.** depicts the elements involved in demonstrating inspection using exported session keys. Both post-facto and real-time decryption capabilities are illustrated.

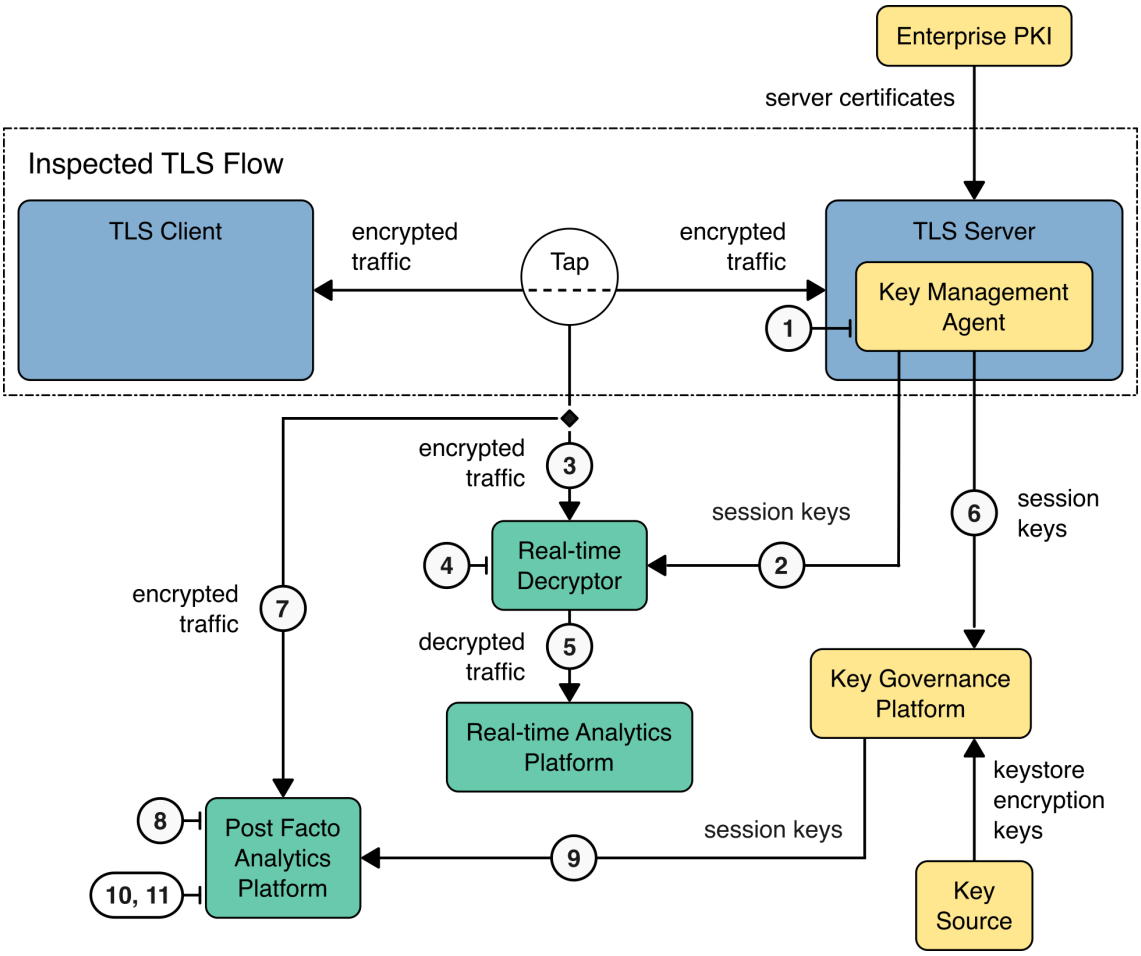

Figure 5-3 Passive Inspection Using Exported Session Keys

5.1.2.1 Real-Time (RT) Decryption

The demonstration of real-time decryption using exported session keys executed the following sequence:

[1] When the TLS Server negotiates a new TLS session with the client, the Key Management Agent sniffs the negotiated session key.
[2] The Key Management Agent sends the session key and client-random-id* to the RT Decryptor.
[3] The Network Tap captures encrypted packets between client and server and forwards them to the RT Decryptor.
[4] The RT Decryptor identifies the flow using the client's random ID and then uses the session key to decrypt the traffic.
[5] The decrypted traffic is forwarded to the RT Analytics Platform.
[6] The Key Management Agent sends the session key and client-random-id to the Key Governance Platform to support post-facto decryption.

*The client-random-id is a plaintext field in the TLS specification that uniquely identifies the TLS session.

Note: Each connection between a server and a client generates a new session key/client-random-id pair. The volume of key material that requires storage can be very large as it increases one-to-one with the volume of traffic. Forward secrecy is maintained.

5.1.2.2 Post-Facto Decryption (follows RT Decryption steps)

The demonstration of decryption for post-facto analysis using exported session keys follows the real-time decryption sequence shown in Section 5.1.5. above:

[7] The Network Tap captures encrypted packets between client and server and forwards them to the Post-Facto Analytics Platform.
[8] The Post-Facto Analytics Platform selects the traffic stream to be decrypted.
[9] On a per-traffic stream basis, the Post-Facto Analytics Platform requests the session key from the Key Governance Platform using the client-random-id from the captured TLS handshake as the lookup key.
[10] The traffic is decrypted using the session key.
[11] Decrypted traffic is available for analysis.

5.1.2.3 Exported Session Key Laboratory Build Components

The software or services used for each of the architecture components in the lab build for this reference architecture can be found in Table 5-2: Build Components for the Passive Decryption Using Exported Session Keys Reference Architecture.

Table 5-2: Build Components for the Passive Decryption Using Exported Session Keys Reference Architecture

Architecture Component	Collaborator	Product Information
Enterprise PKI	DigiCert	• CertCentral Certificate Management Platform
TLS Client		• Mozilla Firefox Browser • Microsoft Edge Browser • Command-line Email Client & Mozilla Thunderbird
Tap		• VMWare vSphere Distributed vSwitch port mirroring
TLS Server(s)		• Containerized NGINX reverse proxy and Keycloak IdAM application • Containerized NGINX reverse proxy and Python test application • Containerized NGINX proxy server • MariaDB Database Server • Containerized Postfix SMTP Server and Dovecot IMAP Server
Key Capture and Registration	Not For Radio	• Encryption Visibility Architecture Encryption Visibility Agent (EVA) Version 0.7.0
Real-time Decryptor	MIRA	• Encrypted Traffic Orchestration (ETO)

		Version 3.1.0-2024.11.28-4781 Virtual Appliance
Key Governance Platform	AppViewX	• AppviewX Version 2020.3.10 build 55 Virtual Appliance
Real-time Analytics Platform	NetScout	• nGeniusONE Version 6.3.5 build 3184 • Omnis Cyber Intelligence (OCI) Version 6.3.5 build 3184
Post-Facto Analytics Platform		• vSTREAM ISNG Version 6.3.5 build 3184 • vCYBERSTREAM ISNG Version 6.3.5 build 3184
Key Source	Thales TCT	• Luna Network HSM T-5000 Firmware: Version 7.11.1 FIPS-validated

5.1.2.4 Installation and Configuration for Exported Session Key Approach

Instructions for the installation and configuration of an exported session key build can be found here.

5.2 Break and Inspect Using Middleboxes

The B&I middlebox architecture supports capturing incoming traffic. It also sends tapped decrypted traffic to an analytics platform, generating traffic re-encrypted using new session keys negotiated between the B&I device and the client and/or server, and passes re-encrypted traffic from the B&I component to an enterprise network server for routing to the enterprise's in-house consumers. The architecture also includes a connection between the server and B&I components to the enterprise PKI. Traffic capture between the client and the B&I device, the B&I device and the server, or both is possible. To enable subsequent analysis of captured encrypted data, provide the session keys between the client and the B&I device, the B&I device and the server, or both, to the key governance platform. With the middlebox (break and inspect) approach, each TLS connection is terminated at the middlebox. The middlebox then initiates a second TLS connection to the target TLS server. The middlebox copies the decrypted TLS payload from the first TLS connection to the second TLS connection while passing the clear text of the traffic to the Real-time Analytics Platform. Finally, the middlebox registers the ephemeral session key for each secondary connection with the Key Governance Platform. The post-facto analytics platform can retrieve the ephemeral keys by querying the Key Governance Platform using the client's random identifier for the TLS session to be decrypted. The descriptive detail for active inspection using middleboxes is provided here.

Figure 4-3 depicts the architectural elements involved in demonstrating visibility using a middlebox. **Note**: Although real-time and post-facto decryption are shown in the architecture drawing, only real-time decryption has been demonstrated as of this writing. Traffic is re-encrypted for transmission to the post-facto traffic capture platform within the data center.

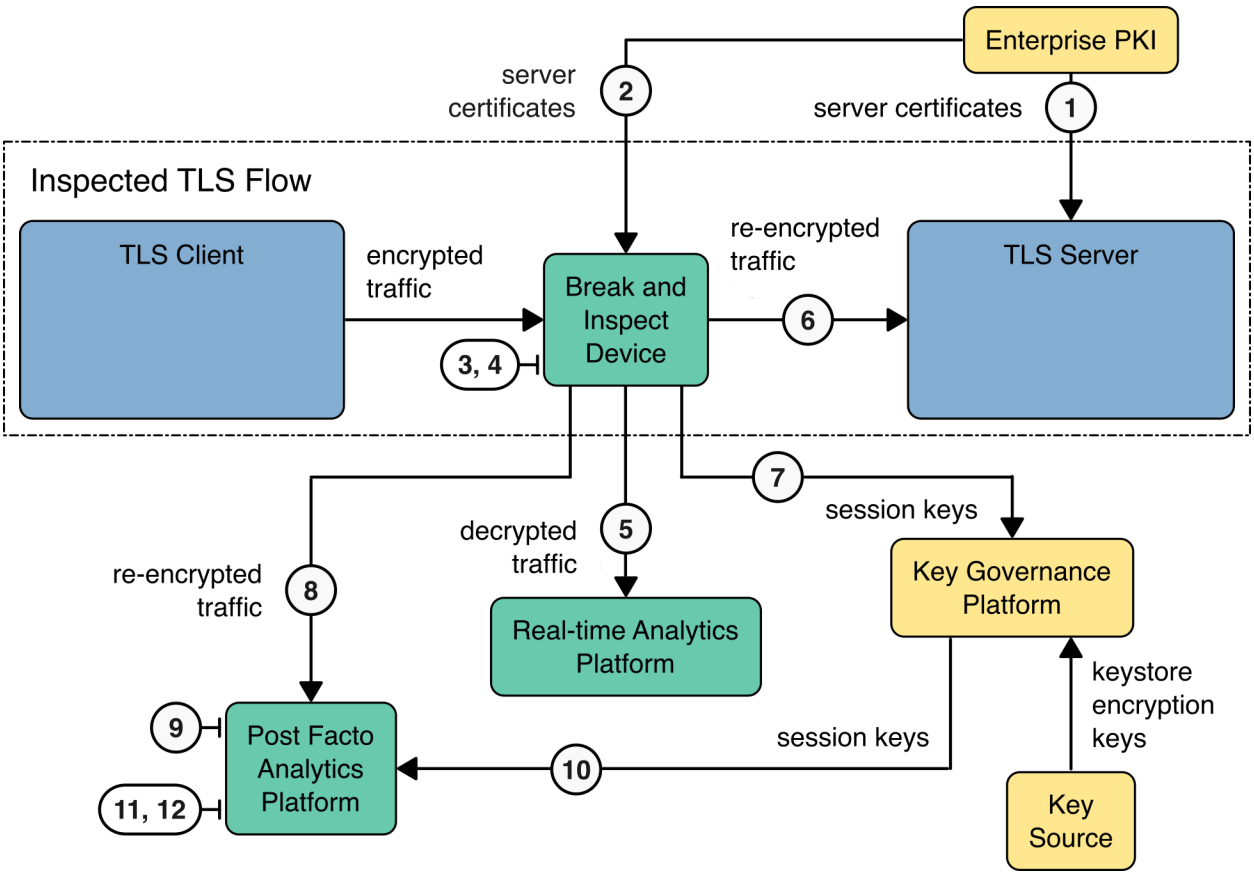

Figure 5-4 Middlebox Break and Inspect Demonstration Elements

5.2.1 Real-Time (RT) Decryption

The real-time break and inspect process executes the following steps:

[1] TLS Server certificates are provisioned on the appropriate TLS Server.
[2] All TLS Server certificates and private keys are loaded into the middlebox.
[3] The TLS client negotiates a TLS session with the middlebox. Simultaneously, the middlebox negotiates a new TLS session with the intended destination TLS Server.
[4] The traffic from the incoming TLS session is decrypted by the middlebox using the session key for the TLS session to the client.
[5] The decrypted traffic is forwarded to the RT Analytics Platform.
[6] The decrypted traffic is copied to the TLS session with the intended destination TLS Server after being encrypted with the session key for this session.
[7] The middlebox exports the session key and client-random-id pair of the TLS session between the middlebox and the intended destination TLS Server to support post-facto decryption. **Note:** The session keys will differ for the client-to-B&I session and the B&I-to-server session. The session keys for the two sessions can be exported.

5.2.2 Post-Facto Decryption (follows RT Decryption steps)

The demonstration of decryption for post-facto analysis using middlebox B&I processes will execute the following sequence that follows the real-time decryption sequence shown above:

[8] The Network Tap captures encrypted packets between the middlebox and server and forwards them to the Post-Facto Analytics Platform.
[9] The Post-Facto Analytics Platform selects the traffic stream to be decrypted.
[10] Per the traffic stream, the Analytics Platform requests the session key from the Key Governance Platform, and the Key Governance Platform provides the session key to the Analytics Platform.
[11] The traffic is decrypted using the session key.
[12] Decrypted traffic is available for analysis.

5.2.3 Middlebox Laboratory Build Components

The lab has two builds of this reference architecture: one that operates on traffic at the OSI Layer 3 level and one that operates on the OSI Layer 2 level.

5.2.3.1 Layer 3 Build

In the lab build for the Layer 3 version of this reference architecture, the software or services used for each of the architecture components are in Table 5-3: Build Components for the Break and Inspect Decryption Reference Architecture (Layer 3 Implementation).

Table 5-3: Build Components for the Break and Inspect Decryption Reference Architecture (Layer 3 Implementation)

Architecture Component	Collaborator	Product Information
Enterprise PKI	DigiCert	• CertCentral Certificate Management Platform
TLS Client		• Mozilla Firefox Browser • Microsoft Edge Browser • Command-line interface, Email Client & Mozilla Thunderbird
Break and Inspect Device (Layer 3)	F5	• BIG-IP SSL Orchestrator Version 17.0.0 build 0.0.22
TLS Server(s)		• Containerized NGINX reverse proxy and Keycloak IdAM application • Containerized NGINX reverse proxy and Python test app • MariaDB Database Server • Postfix SMTP Server and Dovecot IMAP Server
Key Governance Platform	AppViewX	• appviewX Version 2020.3.10 build 55 Virtual Appliance

		• Redis database
Real-time Analytics Platform	NetScout	• nGeniusONE Version 6.3.5 build 3184 • Omnis Cyber Intelligence (OCI) Version 6.3.5 build 3184
Post-Facto Analytics Platform		• vSTREAM ISNG Version 6.3.5 build 3184 • vCYBERSTREAM ISNG Version 6.3.5 build 3184
Key Source	Thales TCT	• Luna Network HSM T-5000 Firmware: Version 7.11.1 FIPS-validated

F5 BIG-IP SSL Orchestrator (SSLO) provides an all-in-one appliance solution designed specifically to optimize the SSL infrastructure, provide security devices with visibility of TLS-encrypted traffic, and maximize the efficient use of existing security resources. The SSL Orchestrator makes encrypted traffic visible to security solutions and optimizes existing security elements. It delivers dynamic service chaining and policy-based traffic steering by applying context-based intelligence to encrypted traffic handling to intelligently manage the flow of encrypted traffic across the security stack.

5.2.3.2 Layer 2 Build

The lab build for the Layer 2 version of this reference architecture uses the software or services for each of the architecture components found in Table 4.4.

Table 5-4: Build Components for the Break and Inspect Decryption Reference Architecture (Layer 2 Implementation)

Architecture Component	Collaborator	Product Information
Enterprise PKI	DigiCert	• CertCentral Certificate Management Platform
TLS Client		• Mozilla Firefox Browser • Microsoft Edge Browser • Command-line interface
Break and Inspect Device (Layer 2)	MIRA	• Encrypted Traffic Orchestration (ETO) Version 3.1.0-2024.11.28-4781 Virtual Appliance
TLS Server(s)		• Containerized NGINX reverse proxy and Keycloak IdAM application • Containerized NGINX reverse proxy and Python test app • MariaDB Database Server • Postfix SMTP Server and Dovecot IMAP Server

Key Governance Platform	AppViewX	• appviewX Version 2020.3.10 build 55 Virtual Appliance • Redis database
Real-time Analytics Platform	NetScout	• nGeniusONE Version 6.3.5 build 3184 • Omnis Cyber Intelligence (OCI) Version 6.3.5 build 3184
Post-Facto Analytics Platform		• vSTREAM ISNG Version 6.3.5 build 3184 • vCYBERSTREAM ISNG Version 6.3.5 build 3184
Key Source	Thales TCT	• Luna Network HSM T-5000 Firmware: Version 7.11.1 FIPS-validated

Mira ETO software supports B&I mode on all types of appliances, physical and virtual (KVM and ESXi), and when deployed in the public cloud (AWS). In this project architecture, the Mira ETO is installed inline and can provide real-time decryption and re-encryption of TLS traffic to maintain an end-to-end TLS connection between the client and the server. Inline interfaces (real or virtual) create a bump in the wire. Decrypted versions of the traffic can be sent to both passive and inline tools. Passive tools consume the decrypted stream and generate alerts. Inline tools process the decrypted stream, then return it (potentially modified) to the ETO for re-encryption before it travels to the client or server. There is a single interface for passive tools and two interfaces for inline tools.

5.2.4 Installation and Configuration for Active Middlebox Approach

5.2.4.1 Installation and Configuration for Layer 3 Build

The instructions for the installation and configuration of the Layer 2 active middlebox build are here.

5.2.4.2 Installation and Configuration for Layer 2 Build

Instructions for the installation and configuration of the Layer 3 active middlebox build can be found here.

5.3 NCCoE Laboratory Physical Architecture

The descriptive detail for the TLS 1.3 Visibility Laboratory Architecture is provided online here.

5.4 Specific Details

Each of the demonstrations identified in Section 5.2 generated the expected results. Specific details of the demonstration results are provided in Appendix F.

6 Functional Demonstrations

6.1 Usage Scenarios Supported

The TLS 1.3 visibility project demonstrates how to enable a variety of security monitoring and analysis activities that support enterprise compliance, security, and operational requirements. Representative scenarios described in this section involve enterprise data center environments that may include on-premises and hybrid cloud deployments hosted by a third-party data center or a public cloud provider. Organizations may need access to plaintext traffic entering their systems for reasons ranging from fraud detection to enforcement of system use policies to cybersecurity monitoring and analysis. Examples of scenarios where visibility into traffic content for security compliance and continuity of operations is required include outbound traffic, connections across the internet to the enterprise network boundary, and communications within the enterprise network between internal systems. This project focuses on communications within the enterprise network and does not focus on outbound connections or communications across the public internet. Some example scenarios requiring visibility into TLS 1.3-protected traffic include troubleshooting, performance monitoring, cybersecurity threat triage, and cybersecurity forensics. Individual enterprises may apply the visibility techniques outlined in this project to selected representative scenarios or to additional scenarios tailored to their specific operations.

6.1.1 Troubleshooting Scenario

If availability and operational problems occur, enterprises that provide these services to customers, partners, and employees need to rapidly troubleshoot and fix these issues. Operations troubleshooting scenarios demonstrate the enterprise tracing transactions through all tiers of an application, including the collection of detailed information such as transaction identifiers, data payload, and the results of operations performed by each application tier. Because operational issues can be intermittent and difficult to replicate, troubleshooting scenarios include the ability to proactively collect and view detailed historical data that may or may not appear in logs. Examples of troubleshooting situations include application unavailability and intermittent system failures. Visibility into enterprise elements such as communications for network-attached storage (NAS), identity management systems, databases, routers and switches, application servers, web servers, load balancers, and firewalls can help provide a complete picture of the end-to-end session across the enterprise. The troubleshooting scenario includes these elements:

- Demonstrating the ability to trace transactions across multiple tiers of applications and communications infrastructure, such as load balancers, firewalls, routers, and switches.
- Collecting detailed information that shows the outcomes of operations performed (which may or may not be recorded in logs).
- Accessing detailed historical data (which may or may not be available in logs).

6.1.2 Performance Monitoring Scenario

Application performance and response times are critical to customer service and time-sensitive mission-critical applications. Performance issues may range from application-specific response degradation to malicious distributed denial of service attacks. Enterprises must proactively detect and isolate performance issues for multi-tier applications.

The performance monitoring scenario includes the following elements:

- Rapidly and accurately detect user performance issues.
- Predict and resolve customer performance issues based on upstream degradation.
- Ability to rapidly identify the source of performance issues.
- Monitor across all mission-critical applications/platforms.
- Minimize performance load on applications/platforms.

6.1.3 Cybersecurity Threat Triage and Forensics Scenario

With the widespread threat of cyber-attacks, enterprises must rapidly triage indicators of compromise (IOCs)—separating false positives from real attacks. The threat triage scenario includes triage, identification, and response to IOCs that may arise in a variety of enterprise elements. Examples include network-attached storage, identity management systems, databases, routers and switches, application servers, web servers, load balancers, and firewalls. IOCs may appear in processes, open ports, and logs. The cybersecurity threat triage scenario includes the following elements:

- Rapidly get a clear picture of the system state.
- Reduce triage time with an accurate, detailed picture of current and historical communications.
- Minimize reliance on data sources that can be manipulated by attackers (including end-point devices).

After a major compromise, enterprises must quickly identify how the attack occurred. This includes pinpointing each compromised system, the exploited vulnerabilities, the attack methods used, and the data exfiltrated. To be effective, obtain accurate information about all operations performed by attackers (even if the attacker manipulated the logs) from independent data sources. The security forensics scenario includes the ability to trace paths of attacks as they pivot laterally across the internal network of compromised systems. Affected systems may involve network-attached storage, identity management systems, databases, routers and switches, application servers, web servers, load balancers, client systems, and firewalls.

The cybersecurity forensics component of this scenario includes the following elements:

- Ability to trace the path of attack across a network of compromised systems.
- Accurate information about all operations performed by attackers.
- Facilitate the creation of updated and improved operations and security policies to prevent or mitigate the success of future attacks.

6.1.4 Monitoring for Compliance and Hygiene Scenario

Enterprises that conduct proactive traffic assessments can use these traffic assessments as a baseline for comparison of the attributes of the traffic being monitored against their forward-looking expectations for cybersecurity. The compliance and hygiene scenario shows how to verify that the observed traffic profile complies with current cybersecurity standards and objectives. This includes verifying that updates are implemented according to enterprise policies and detecting instances where disallowed security components, outdated protocols, systems, hardware, or software versions are in use.

6.2 Example Demonstration Events

Twelve events were chosen to demonstrate the scenarios outlined in Section 6.1. Table 6.1 lists these demonstration events.

Table 6-1: Demonstration Events

Troubleshooting Examples
Identification of Failed Network Traffic Due to Expired TLS PKI Certificates (Layer 4)
Protocol-Specific Service Utilization and Consumption Characteristics Identification and Logging
Identification of and Reporting Protocol-Specific Error Status Codes
Performance Monitoring Examples
Identification of, Collection of, and Reporting on Protocol-specific Error Status Codes for Services
Identification of Propagation of Performance Issues Throughout a System by Correlating Error Status Codes Across Component Services from TLS Connections on Either Side of a Proxy
Develop Baselines for Traffic Performance Characteristics for Individual Servers
Threat Triage and Forensics Examples
Scan for Suspicious/Malicious Content
Detection of Unanticipated Inability to Decrypt Traffic
Detection of Command-and-Control and Exfiltration Activity
Scanning of Network Traffic for Un-sanitized User Input
Monitoring for Compliance and Hygiene Examples
Detection of the Use of Outdated Protocols
Detection of, Identification of, and Reporting on the Use of Outdated Software

These events demonstrate a build's capability to operate in each of the scenarios presented in Section 5.1. Specific product configurations for the examples are included in Appendix D on the NIST pages site. Demonstration input and output information, traffic generated, results, and screenshots associated with the validation of examples are posted in Appendix E on the NIST pages site.

7 Risk and Compliance Management

NIST SP 800-30 Revision 1, *Guide for Conducting Risk Assessments*, [11] states that risk is "a measure of the extent to which an entity is threatened by a potential circumstance or event, and typically a function of: (i) the adverse impacts that would arise if the circumstance or event occurs; and (ii) the likelihood of occurrence." The guide further defines risk assessment as "the process of identifying, estimating, and prioritizing risks to organizational operations (including mission, functions, image, reputation), organizational assets, individuals, other organizations, and the Nation, resulting from the operation of an information system. Part of risk management incorporates threat and vulnerability analyses and considers mitigations provided by security controls planned or in place."

The NCCoE recommends that any discussion of risk management—particularly for an organization with a defined mission/goal and a defined boundary, using systems to execute that mission, and with responsibility for managing its own risks and performance—should begin by reviewing NIST SP 800-37 Revision 2, *Risk Management Framework for Information Systems and Organizations*, [12] material that is publicly available. The Risk Management Framework (RMF) [13] is invaluable and gave us a baseline to assess risks, from which we developed the project, the security characteristics of the build, and this guide.

7.1 Threats

General threats to data exchanged over networks include eavesdropping, tampering, and forgery. TLS uses cryptographic mechanisms to protect data from being stolen, modified, or spoofed. This section describes generic threats to the security of information that TLS mechanisms protect.

- **Stolen keys**. If threat actors gain unauthorized access to symmetric keys used for data encryption or private keys used in public key cryptography, they can bypass authentication systems or gain access to other keys. If symmetric or private keys are stored without encryption, they are especially at risk of being exploited to undermine the security protections that TLS provides.
  - Inadequately protected keys are subject to unauthorized access by direct theft, hacking, or other means. The more data a key protects, the more severe the consequences typically are if a threat actor gains unauthorized access.
  - TLS 1.3 symmetric keys are used with Authenticated Encryption with Associated Data (AEAD) algorithms—providing confidentiality and integrity for the keys.
- **Certificate compromise**. Public keys used during authentication are often exchanged in certificates issued by a CA. A real threat exists if the issuing CA is compromised or if the registration system, persons, or process are used to obtain an unauthorized certificate in the name of a legitimate entity to compromise the clients.
- **Handshake data replay**. Parties to cryptographically protected communications like TLS exchange keys use protocols often called "handshakes," which often include multiple steps. TLS 1.3 allows the client to send data (known as 0-RTT data) in the first flight of a handshake with a server that previously connected to the client. Replayable 0-RTT data presents several security threats to TLS-using applications unless those applications are specifically engineered to be safe under replay (minimally, an HTTP method that is idempotent, but could require stronger conditions, such as constant-time response). Many applications do not allow 0-RTT to avoid the replay concern (e.g., draft-ietf-netconf-over-tls13).

- **Potential attacks on the security** of information that TLS mechanisms protect include:
  - Duplicating actions that cause side effects (e.g., purchasing an item or transferring money) to be duplicated, thus harming the site or the user.
  - Storing and replaying 0-RTT messages to reorder them with respect to other messages (e.g., moving a delete to after a create).
  - Exploiting cache timing behavior to discover the content of 0-RTT messages by replaying a message to a different cache node and then using a separate connection to measure request latency to see if the two requests address the same resource.
  - If data is replayed a large number of times, additional attacks become possible, e.g., repeated measurements of cryptographic operation speed. In addition, they can overload rate-limiting systems.
- **Misuse of client credentials** residing on the client machine.
- **Absence of support for endpoint solutions**. Not all server-type systems within an enterprise are supported by endpoint vendors. Enterprises commonly have old systems running custom operating system (OS) software for which there is limited or no support in endpoint solutions.
- **Systems compromised by running endpoint software**. When deployed, endpoint solutions may provide good security up to the point where the system running the endpoint or endpoint software is compromised. An endpoint solution may not detect an endpoint compromise if it relies on trusting what the endpoint software tells it, resulting in false trust in the endpoint. Alternate solutions that analyze network traffic can detect compromised or rogue endpoints and are resilient when used alongside endpoint solutions.

7.2 Vulnerabilities

Several vulnerabilities were found in both the TLS 1.2 protocol and in the implementation of features permitted by TLS 1.2. While TLS 1.2 can be made secure using extensions and careful configuration, the design of TLS 1.3 removes vulnerabilities that exist when using TLS 1.2. Vendors are taking note and focusing on TLS 1.3. Therefore, getting new algorithms or extensions for TLS 1.2 for implementation by vendors is increasingly difficult. Below are some examples of the challenges when moving to TLS 1.3:

- Unlike TLS 1.3, TLS 1.2 offers some cipher suites, such as those that use RSA key exchange, that do not provide forward secrecy. Where forward secrecy is not provided, if a TLS-enabled server is compromised, the contents of its previous TLS communications are vulnerable to exposure. The ephemeral key exchange mechanisms that provide forward secrecy also protect future TLS communication against passive attackers.
- Enterprises widely use passive decryption techniques to achieve visibility into their internal TLS 1.2 enterprise traffic. These techniques only work with the cipher suites that use RSA key exchange. The RSA key exchange used in TLS 1.2 does not provide forward secrecy—making it vulnerable to a number of implementation flaws. As a result, its use has been deprecated. (See SP 800-131A Rev. 2 [14], SP 800-52 Rev. 2 [15], IETF draft Deprecating Obsolete Key Exchange Methods in TLS 1.2 [16], and TLS 1.2 Is Frozen [17].)
- Reusing keys outside the protected data center creates vulnerabilities around the comparison of key shares in different handshakes. These vulnerabilities may enable an attacker to track an endpoint or reveal the identity of the TLS server to which a user is connected. On the public internet, this is a violation of user privacy. The current TLS 1.3 specification contains a new normative requirement stating that to prevent tracking and identification, “Clients SHOULD NOT reuse

a ticket for multiple connections" (RFC 8446 section C.4. [1]). Reuse of a key share allows passive observers to correlate different connections. This specification discourages client and server reuse of a key share for multiple internet connections. Reusing key shares outside protected facilities can also expand the impact of security breaches.

- Except in cases of exclusively symmetric key management environments, sharing symmetric keys should be restricted to data center environments. A node with access to the symmetric traffic keys can view all traffic and impersonate the endpoints by modifying and injecting traffic.
- Some TLS 1.2 visibility mechanisms that are based on RSA private key sharing allow middleboxes to masquerade as servers. The TLS 1.3 mechanisms prevent this because it is unnecessary to share the signature private key to gain visibility.

7.3 Risk

TLS 1.3 offers improved performance and efficiency and more robust security than TLS 1.2. TLS 1.3's more robust security includes mandating forward secrecy resulting in a capability gap for enterprises whose IT security teams have been able to meet requirements for visibility into TLS 1.2 network traffic. From a cryptographic point of view, enterprises implementing TLS 1.3 might effectively mandate a reliance on endpoint solutions to achieve their operational security requirements. However, this can be impractical for many enterprises.

For instance, if an enterprise relied entirely upon endpoint security—without any visibility by middleboxes—steps would need to be taken that require time and resources that could potentially impact an enterprise's migration to TLS 1.3.

A frequent first step taken by a sophisticated attacker on a target is to modify, disable, or evade endpoint security tools. This is an enduring reality faced by "blue team security practitioners" (e.g., security operations or incident response teams) for decades. Researchers continue to publish results of successful evasion and neutering techniques for all endpoint controls, including the most sophisticated tools available on the market. Nonetheless, if malware has detonated on a computer, the only thing between it and the endpoint security tool is the OS. To combat this risk, organizations should budget for and perform the following steps:

- Follow OS hardening requirements to limit the ability of software-accessible accounts to control endpoint security tools.
- Implement strategies that can help identify missing, fraudulent, or anomalous status or log data from the endpoint tool(s).
- Upgrade, modernize, or replace all legacy applications that leverage insecure libraries, protocols, applications, and subsystems throughout the endpoint state.
- Ensure that remote access to all endpoint agent management tools adheres to the most stringent authentication/authorization strategies.
- Implement strategies to persistently refresh endpoint agent policy or accurately detect policy drift.
- Implement a credible application allowlist solution to prevent execution/reading of unauthorized applications and libraries.
- Implement endpoint control techniques such as enhanced logging as effective detective controls where the following conditions are met:

    - Secure adequate funding to maintain a robust security information and event management (SIEM) infrastructure
    - Make trained personnel available to manage an SIEM or similar tool to create and administer correlation rules to produce timely, accurate alerts related to anomalous activity.
- Implement appropriate configuration of all cogent log-generating agents on the endpoint (e.g., instant log transfer to SIEM vs. periodic transfer of batched logs that can be deleted by an attacker to hide their activity).

See NIST SP 800-92 Revision 1, Cybersecurity Log Management Planning Guide, [18] for more information on logging and log protection.

Different types of malware execute with different intentions. Destructive malware often doesn't require the instantiation of command-and-control (C2) communication with attacker infrastructure. However, the goals of the most sophisticated adversaries (e.g., financial gain, data exfiltration) are best achieved by maintaining persistence via a C2 channel over a network to the attacker. To counter this action, organizations should make security controls surrounding outbound network communications from organizational endpoints maximally restrictive.

The above concepts could be beneficial to an organization, but may not be realizable.

As stated previously, re-architecting networks is difficult, expensive, and time-consuming. Even if viable, it is not a practical short-term solution for any organization, including large data centers. To mitigate risk, organizations must make their endpoint solutions stable, capable, and effective. To meet these goals, endpoint solutions must become consistent and reliable recorders of all related events (enhanced logging). In addition, producing, collecting, storing, and parsing terabytes (or more) of data requires building a completely separate infrastructure. This is not a simple proposition. Building this infrastructure requires DPI for certain data, as well as for management activities. Furthermore, if the true root cause of a compromise or other network problem is occurring at a middlebox device, endpoints will not see this information. Using intermediate proxies between all tiers adds cost, latency, and potential points of failure. It also becomes less viable as the number of tiers per given application increases; costs and complexity arise, too, in many cases. Finally, situations will occur where intermediate proxies are not possible, such as secure subnets and virtual environments.

When adopting any visibility solution, it's imperative to protect stored session keys from being accessed by external entities to the data center. This requires implementing access controls that enforce least privilege within the data center. Consequently, management's risk assessment involves trading off the relative consequences of delaying TLS 1.3 implementation or replacing enterprise data centers against the cost of protecting stored session keys from access by processes other than those that require or are approved for specific continuous monitoring or forensics functions. While implementing visibility solutions, enterprises must focus on supporting least privilege principles, compliance with zero trust, and supply chain security requirements. See NIST SP 800-207 [19]. Ideally, implementations will provide network owners with control over what information is actually shared with monitoring systems. Cryptographic protection of all keys, whether at rest or in transit, is essential except where the keys are being employed in a cryptographic process or approved compensating controls are employed.

Standardized open interfaces for endpoint interception are needed but are not currently available in applications that scale to the requirements of large data centers. As a result, organizations may

determine that acceptance of some cryptographic security risks associated with the visibility solutions described herein is acceptable in the face of the consequences of delaying TLS 1.3 implementation and of loss of visibility into information exchanges by the IT security staff responsible for security monitoring and forensics. One of the visibility approaches demonstrated in this project involves middlebox solutions. The risks associated with introducing in-house middleboxes that may have vulnerabilities enabling network attacks, where an attacker intercepts communications and/or caching or otherwise storing session keys, may be weighed against the consequences of losing security monitoring and forensics capabilities.

7.4 Security Control Map

Any organization can use security control mappings to implement or refine TLS 1.3 visibility solutions. The mappings explain how cybersecurity functions from the reference design relate to NIST-recommended security outcomes and controls. See the security outcome subcategories from the NIST Cybersecurity Framework (*Framework for Improving Critical Infrastructure Cybersecurity* [CSF] 2.0) [20] and security controls identified in NIST SP 800-53r5 (*Security and Privacy Controls for Information Systems and Organizations*) [21] for more details. All of the elements in these mappings—the TLS 1.3 visibility cybersecurity functions, CSF Subcategories, and SP 800-53 controls—are concepts involving ways to reduce cybersecurity risk. The mapping methodology is described in NIST IR 8477 *Mapping Relationships Between Documentary Standards, Regulations, Frameworks, and Guidelines: Developing Cybersecurity and Privacy Concept Mappings* [22].

This mapping helps answer the following questions:

1. **Why should organizations implement TLS 1.3 visibility solutions?** This use case identifies how implementing TLS 1.3 visibility solutions can help organizations achieve CSF Subcategories and SP 800-53 controls. Communicating to those individuals who expend resources to implement TLS 1.3 visibility, e.g., chief information security officers and security teams, can help fulfill other security requirements.
2. **How can organizations implement TLS 1.3 visibility solutions?** This use case identifies how an organization's existing implementations of CSF Subcategories and SP 800-53 controls can help support the trusted implementation of TLS 1.3 visibility solutions. An organization looking to implement TLS 1.3 visibility solutions might first assess its current security capabilities so that it can plan how to add missing capabilities and enhance existing capabilities. Organizations can leverage their existing security investments and prioritize future security technology deployment to address the gaps.

Mappings between cybersecurity functions performed by the reference design's logical components and the security characteristics enumerated in relevant cybersecurity documents are available here.

8 Demonstration and Future Considerations

8.1 General Findings and Observations

Each of the demonstrations identified in Section 5.2 generated the expected results. Specific details of the demonstration results are provided in Appendix F.

8.2 Future Build Considerations

8.2.1 Planning for Visibility with Post-Quantum Cryptography (PQC)

Post-Quantum Cryptography (PQC) includes digital signature algorithms and key encapsulation mechanisms (KEM). The TLS 1.3 visibility solutions don't change key management, signature generation, or signature verification processing. Therefore, the future use of PQC signature algorithms will not impact the applicability of these visibility solutions.

Likewise, TLS 1.3 visibility solutions that store session keys are not impacted by the transition to PQC algorithms.

In the future, replacing the Diffie-Hellman algorithm with a PQC KEM will have some impact on the visibility solutions. Figure 8-1 shows the most likely use of a KEM in the TLS 1.3 handshake. Note: The only secret key is held by the client. In the TLS 1.3 visibility solutions, the servers are responsible for storing the short-lived Diffie-Hellman private keys.

Client:

KEM. KeyGen() -> (pk, sk)

Sent the public key (pk) to the server as part of the ClientHello

Server:

KEM.Encapsulate(pk) -> (ct, ss)

Use the shared secret (ss) as an input to the key schedule

Sent the ciphertext (ct) to the client in the ServerHello

Client:

KEM.Decapsulate(sk, ct) -> ss

Use the shared secret (ss) as an input to the key schedule

Figure 8-1 Sample use of PQC KEM in TLS 1.3 Handshake

8.2.2 Client-Based Monitoring

Client-based monitoring is an additional capability that is not within the scope of the current demonstration project. It may be included in future project extensions. The project examined server-focused options rather than client-focused ones. Another approach to TLS 1.3 visibility involves reliance on enterprise support directly by the client endpoint or using clients via trusted proxy methods (e.g., SOCKS proxies). This approach is reported to require no potential deviation from RFC 8446. It also ensures that only those TLS clients mandated by local policy (e.g., enterprise management, parental control, anti-malware protection services) have these monitoring features available, and those only via opt-in (directly or via their guardian).

Appendix A Glossary

We use the terms from NISTIR 7298, *Glossary of Information Security Terms* [23] or IETF RFC 4949, *Internet Security Glossary*, Version 2 [24], where those references define the terms.

Analytics	The discipline that applies logic and mathematics to data to provide insights for event recognition and for making response decisions. In this project, the function is executed by a set of tools for examining unencrypted payloads to identify undesired characteristics.
Bounded-Lifetime Key	A key variable that is used within the enterprise for decryption in real time or is stored for a period established by an explicit enterprise policy to enable decryption for post-facto security analytics/forensics purposes and is then destroyed in accordance with the policy.
Break and Inspect	A function that taps, decrypts, terminates, and re-encrypts/reinitiates network traffic.
Certificate	A set of data that uniquely identifies a public key (which has a corresponding private key) and an owner that is authorized to use the key pair. The certificate contains the owner's public key and possibly other information and is digitally signed by a certificate authority (i.e., a trusted party), thereby binding the public key to the owner.
Certificate Authority	An authorized entity that stores, signs, and issues digital cryptographic key certificates. It acts to validate identities and bind them to cryptographic key pairs with digital certificates.
Certificate and Key Governance	Functions for securely issuing, monitoring, facilitating, and executing digital X.509 certificates and managing the cryptographic keys exchanged using the certificates.
Client	System entities that request and use a service provided by another system entity called a server. Usually, it is understood that the client and server are automated components of the system, and the client makes the request on behalf of a human user. Clients may initiate encrypted traffic. They are interfaces for human users, devices, applications, and processes to access network functions, including the requesting of certificates and keys.
Cryptography	The discipline that embodies the principles, means, and methods for the transformation of data to hide their semantic content, prevent their unauthorized use, or prevent their undetected modification. It embodies the principles, means, and methods for providing information security, including confidentiality, data integrity, non-repudiation, and authenticity.
Decryption	The process of a confidentiality mode that transforms encrypted data into the original usable data.

Deep Packet Inspection	A form of packet filtering that locates, identifies, classifies, and reroutes or blocks packets with specific data or code payloads that conventional packet filtering, which examines only packet headers, cannot detect.
DevOps	A combination of the terms development and operations, meant to represent a collaborative or shared approach to the tasks performed by a company's application development and IT operations teams.
Diffie-Hellman	A method used to securely exchange or establish secret keys across an insecure network. Ephemeral Diffie-Hellman is used to create temporary or single-use secret keys.
Encryption	Cryptographic transformation of data (called “plaintext”) into a form (called “ciphertext”) that conceals the data’s original meaning to prevent it from being known or used. If the transformation is reversible, the corresponding reversal process is called “decryption,” which is a transformation that restores encrypted data to its original state.
Endpoint Agent	A lightweight background application installed on a device's operating system to constantly assess it for vulnerabilities.
Ephemeral Key	A cryptographic key that is generated for each execution of a key-establishment process and that meets other requirements of the key type (e.g., unique to each message or session).
Key	A numerical value used to control cryptographic operations, such as decryption, encryption, signature generation, or signature verification. Usually, a sequence of random or pseudorandom bits used initially to set up and periodically change the operations performed in cryptographic operations for the purpose of encrypting or decrypting electronic signals or for producing another key.
Key Capture	Captures of session keys at the time they are negotiated.
Key Management	The handling of cryptographic keys and other related security parameters (e.g., passwords) during the entire life cycle of the keys, including their generation, storage, establishment, entry and output, and destruction.
Key Registration	A function in the lifecycle of a cryptographic key; the process of a registration authority officially recording the keying material.
Key Source	A FIPS 140-validated entity that securely generates cryptographic keys and key pairs that are used in cryptography.
Kubernetes	A portable, extensible, open-source platform for managing containerized workloads and services that facilitates both declarative configuration and automation.
Middlebox	A networking device that transforms, inspects, filters, and manipulates traffic for purposes other than packet forwarding. In this project, the device is used to break and inspect enterprise network traffic.

Network Tap	A component that provides a copy of traffic from a network segment. It is typically used in network security applications to monitor traffic and identify malicious activity or security threats.
Post-Facto	From or by an after act, or thing done afterward; in consequence of a subsequent act; retrospective.
Private Key	The secret part of an asymmetric key pair that is typically used to digitally sign or decrypt data.
Public Key	The public part of an asymmetric key pair that is typically used to verify signatures or encrypt data.
Public Key Certificate	A digital document issued and digitally signed by the private key of a certificate authority that binds an identifier to a cardholder through a public key. The certificate indicates that the cardholder identified in the certificate has sole control and access to the private key.
Public Key Infrastructure	A framework that is established to issue, maintain, and revoke public key certificates.
QUIC	A UDP-based multiplexed and secure transport protocol.
Real-Time	A function that conducts operations that must guarantee response times within a specified time or window of time, usually relatively short.
SecOps	A combination of the terms security and operations; a methodology that IT managers implement to enhance the connection, collaboration, and communication between IT security and IT operations teams.
Secret Key	A cryptographic key that is used with a (symmetric) cryptographic algorithm that is uniquely associated with one or more entities and is not made public. The use of the term “secret” in this context does not imply a classification level but rather implies the need to protect the key from disclosure.
Server	A system entity that provides services in response to requests from other system entities called clients.
Symmetric Cryptography	A cryptographic algorithm that uses the same secret key for its operation and, if applicable, for reversing the effects of the operation (e.g., an AES key for encryption and decryption).
Transport Layer Security	A security protocol providing privacy and data integrity between two communicating applications.
TLS Server	The counterparty for encrypted traffic that generates session keys, negotiates encryption protocols, and connects to key management infrastructure.

Appendix B List of Acronyms

AEAD	Authenticated Encryption with Associated Data
API	Application Programming Interface
ASI	(NETSCOUT) Adaptive Session Intelligence
AWS	Amazon Web Services
B&I	Break and Inspect
C2	Command and Control
CA	Certificate Authority
CFIUS	Committee on Foreign Investments in the United States
CLM	Certificate Lifecycle Management
CMS	(Mira) Central Management System
CRADA	Cooperative Research and Development Agreement
CSF	Cybersecurity Framework
DCSA	Defense Counterintelligence and Security Agency
DDoS	Distributed Denial of Service
DH	Diffie-Hellman
DMZ	Demilitarized Zone
DNS	Domain Name System
DoH	DNS over HTTPS
DoQ	DNS over QUIC
DoT	DNS over TLS
DPI	Deep Packet Inspection
DTLS	Datagram Transport Layer Security
ESXi	VMware Purpose-Built Bare Metal Hypervisor
ETO	(Mira) Encrypted Traffic Orchestrator
EVA	(NFR) Encryption Visibility Agent, (NFR) Encryption Visibility Architecture
FIPS	Federal Information Processing Standard
FOCI	Foreign Ownership, Control, or Influence
FS	Forward Secrecy
Gbps	Gigabits per second
HSM	Hardware Security Module

HTTPS	Hypertext Transfer Protocol Secure
IETF	Internet Engineering Task Force
IoC	Indicators of Compromise
IT	Information Technology
ITL	Information Technology Laboratory
JSON	JavaScript Object Notation
KVM	Kernel-based Virtual Machine
LDAP	Lightweight Directory Access Protocol
NCCoE	National Cybersecurity Center of Excellence
NDR	Network Detection and Response
NFR	Not for Radio
NIST	National Institute of Standards and Technology
NISTIR	NIST Internal or Interagency Report
OS	Operating System
PKI	Public Key Infrastructure
RBAC	Role-Based Access Control
REST	Representational State Transfer
RFC	Request for Comments
RMF	Risk Management Framework
RSA	Rivest, Shamir, Adleman
RT	Real Time
RTT	Real-Time Text
SaaS	Software-as-a-Service
SAML	Security Assertion Markup Language
SIEM	Security Information and Event Management
SMTP	Simple Mail Transfer Protocol
SOAR	Security Orchestration, Automation, and Response
SP	Special Publication
SSLO	(F5) SSL Orchestrator
SSO	Single Sign-On
TLS	Transport Layer Security
TMOS	(F5) Traffic Management Operation System

UDP	User Datagram Protocol
vETO	(Mira) Virtual Encrypted Traffic Orchestrator
VM	Virtual Machine
WebUI	Web User Interface
XDR	Extended Detection and Response

Appendix C References

[1] E. Rescorla, *The Transport Layer Security (TLS) Protocol Version 1.3*, Internet Engineering Task Force (IETF) Request for Comments (RFC) 8446, August 2018. Available at https://datatracker.ietf.org/doc/rfc8446/

[2] T. Dierks and E. Rescorla, *The Transport Layer Security (TLS) Protocol Version 1.2*, Internet Engineering Task Force (IETF) Request for Comments (RFC) 5246, August 2008 (Updated October 2015). Available at https://datatracker.ietf.org/doc/rfc5246/

[3] Center for Cybersecurity Policy and Law, Enterprise Data Center Transparency and Security Workshop Report, November 2019. Available at https://www.centerforcybersecuritypolicy.org/insights-and-research/enterprise-data-center-transparency-and-security-workshop-summary-report

[4] National Institute of Standards and Technology, *Virtual Workshop on Challenges with Compliance, Operations, and Security with TLS 1.3*, September 2020. Available at https://www.nccoe.nist.gov/get-involved/attend-events/virtual-workshop-challenges-compliance-operations-and-security-tls-13

[5] Z. Hu, L. Zhu, J. Heidemann, A. Mankin, D. Wessels, and P. Hoffman, *Specification for DNS over Transport Layer Security (TLS)*, Internet Engineering Task Force (IETF) Request for Comments (RFC) 7858, May 2016. Available at https://datatracker.ietf.org/doc/rfc7858/

[6] P. Hoffman and P. McManus, *DNS Queries over HTTPS (DoH)*, Internet Engineering Task Force (IETF) Request for Comments (RFC) 8484, October 2018. Available at https://datatracker.ietf.org/doc/rfc8484/

[7] C. Huitema, S. Dickinson, and A. Mankin, *DNS over Dedicated QUIC Connections*, Internet Engineering Task Force (IETF) Request for Comments (RFC) 9250, May 2022. Available at https://datatracker.ietf.org/doc/rfc9250/

[8] B. Carpenter and S. Brim, *Middleboxes: Taxonomy and Issues*, Internet Engineering Task Force (IETF) Request for Comments (RFC) 3234, February 2002. Available at https://datatracker.ietf.org/doc/rfc3234

[9] U.S. Department of Commerce, *Security Requirements for Cryptographic Modules*, Federal Information Processing Standard (FIPS) 140-3, March 2019. https://doi.org/10.6028/NIST.FIPS.140-3

[10] E. Barker, L. Chen, A. Roginsky, A. Vassilev, and R. Davis, Recommendation for Pair-wise Key-establishment Schemes Using Discrete Logarithm Cryptography, NIST Special Publication (SP) 800-56A Revision 3, April 2018. https://doi.org/10.6028/nist.sp.800-56ar3.

[11] Joint Task Force Transformation Initiative, *Guide for Conducting Risk Assessments*, NIST Special Publication (SP) 800-30 Revision 1, September 2012. https://doi.org/10.6028/NIST.SP.800-30r1

[12] Joint Task Force, *Risk Management Framework for Information Systems and Organizations*, NIST Special Publication (SP) 800-37, December 2018. https://doi.org/10.6028/nist.sp.800-37r2.

[13] Joint Task Force, *NIST Risk Management Framework RMF*. Available at https://csrc.nist.gov/projects/risk-management/about-rmf.

[14] E. Barker and A. Roginsky, *Transitioning the Use of Cryptographic Algorithms and Key Lengths*, NIST Special Publication (SP) 800-131A Revision 2, March 2019. https://doi.org/10.6028/NIST.SP.800-131Ar2

[15] K. McKay and D. Cooper, *Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations*, NIST Special Publication (SP) 800-52 Revision 2, August 2019. https://doi.org/10.6028/NIST.SP.800-52r2

[16] C. Bartle and N. Aviram, *Deprecating Obsolete Key Exchange Methods in TLS 1.2*, Internet Engineering Task Force (IETF), March 2023. Available at https://www.ietf.org/archive/id/draft-ietf-tls-deprecate-obsolete-kex-02.html

[17] R. Salz and N. Aviram, *TLS 1.2 is Frozen,* Internet Engineering Task Force (IETF), June 2023. Available at https://www.ietf.org/archive/id/draft-rsalz-tls-tls12-frozen-01.html

[18] Scarfone, Karen, and M. Souppaya, *Cybersecurity Log Management Planning Guide*, NIST Special Publication (SP) 800-92 Revision 1, Initial Public Draft, October 11, 2023. https://doi.org/10.6028/nist.sp.800-92r1.ipd.

[19] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, *Zero Trust Architecture*, NIST Special Publication (SP) 800-207, August 2020. https://doi.org/10.6028/NIST.SP.800-207

[20] National Institute of Standards and Technology, *The NIST Cybersecurity Framework (CSF) 2.0*, February 26, 2024. https://doi.org/10.6028/nist.cswp.29.

[21] Joint Task Force, *Security and Privacy Controls for Information Systems and Organizations*, NIST Special Publication (SP) 800-53 Revision 5, September 2020. https://doi.org/10.6028/NIST.SP.800-53r5

[22] K. Scarfone, M. Souppaya, and M. Fagan, *Mapping Relationships Between Documentary Standards, Regulations, Frameworks, and Guidelines: Developing Cybersecurity and Privacy Concept Mappings*, National Institute of Standards and Technology Interagency Report (NISTIR) 8477, February 2024. https://doi.org/10.6028/NIST.IR.8477

[23] C. Paulson and R. Byers, *Glossary of Key Information Security Terms*, National Institute of Standards and Technology Interagency Report (NISTIR) 7298 Rev. 3, July 2019. https://doi.org/10.6028/NIST.IR.7298r3

[24] R. Shirey, *Internet Security Glossary*, *Version 2,* Internet Engineering Task Force (IETF) Request for Comments (RFC) 4949, August 2007. Available at https://datatracker.ietf.org/doc/rfc4949/

Appendix D Description of the Architectures

This architectural descriptive detail is available here.

D.1 Passive Inspection using Bounded-lifetime DH Server Keys

This passive inspection architectural descriptive detail for bounded lifetime DH is available here.

D.2 Passive inspection using Exported Session Keys

This passive inspection architectural descriptive detail for exported session keys is available here.

D.3 Active Inspection using a Break-and-Inspect Middlebox

This active inspection architectural descriptive detail for middlebox use is available here.

Appendix E Descriptions of the Build Implementations

This build implementation detail is available here.

E.1 Shared Components Across All Builds

This descriptive detail for components shared across all builds is available here.

E.2 Implementation of the Bounded Lifetime DH Key Architecture

This descriptive detail for the implementation of the bounded lifetime DH key architecture is available here.

E.3 Implementation of the Exported Session Key Architecture

This descriptive detail for the implementation of the exported session key architecture is available here.

E.4 Implementation of Middlebox Architecture Implementations

This architectural descriptive detail for middlebox use is available here.

This descriptive detail for the implementation of the middlebox architecture at the OSI Layer 2 is available here.

This descriptive detail for the implementation of the middlebox architecture at the OSI Layer 3 is available here.

Appendix F Details of the Functional Demonstrations and Results

See here for functional demonstrations detail.

F.1 Traffic Visibility to Support Troubleshooting

See here for troubleshooting demonstration detail.

F.2 Traffic Visibility to Support Performance Monitoring

See here for performance monitoring demonstration detail.

F.3 Traffic Visibility to Support Cybersecurity Threat Triage and Forensics

See here for threat triage and forensics demonstration detail.

F.4 Traffic Visibility to Support Monitoring for Compliance and Hygiene

See here for monitoring for compliance and hygiene demonstration detail.

F.5 Functional Demonstration Scripts and Results

See here for demonstration scripts and results.

F.5.1 Scenario 1.1 – Identify Failed Network Traffic Due to Expired TLS PKI Certificates (Layer 4)

See here for Scenario 1.1 script and results.

F.5.2 Scenario 1.2 – Identify and Log Protocol-Specific Distinct Characteristics of Layer 5, 6, and 7-type Service Utilization and Consumption Information

See here for Scenario 1.2 script and results.

F.5.3 Scenario 1.3 – Identify, Collect, and Report on Protocol-Specific Error Status Codes for Services (Layer 5, 6, and 7-type status codes)

See here for Scenario 1.3 script and results.

F.5.4 Scenario 2.1 – Identify, Collect, and Report on Protocol-Specific Error Status Codes for Services.

See here for Scenario 2.1 scripts and results for Layer 2 demonstration and Layer 3 Demonstration.

F.5.5 Scenario 2.2 – Identify the Propagation of Performance Issues Throughout a System by Correlating Error Status Codes Across Component Services

See here for Scenario 2.2 script and results.

F.5.6 Scenario 2.3 – Develop Baselines for Traffic Performance Characteristics for Each Server

See here for Scenario 2.3 script and results.

F.5.7 Scenario 3.1 – Scan Network Flows Content for Malware

See here and here for Scenario 3.1a and 3.1b script and results.

F.5.8 Scenario 3.2 – Scan Network Traffic for Unauthorized Encrypted Connections (i.e., unexpected encryption types, unauthorized encryption protocols, unencrypted traffic, traffic that can't be decrypted, etc.)

See here for Scenario 3.2 script and results.

F.5.9 Scenario 3.3 – Scan Network Traffic Content for Known Command-and-Control or Exfiltration Protocols

See here for Scenario 3.3 script and results.

F.5.10 Scenario 3.4 – Scan Network Traffic for Un-Sanitized User Input

See here for Scenario 3.4 script and results.

F.5.11 Scenario 4.1 – Identify and Report on the Use of Outdated Protocols (and/or 'practices')

See here for Scenario 4.1 script and results.

See here for Scenario 4.2 script and results.

Appendix G Appendix G: Security Control Mapping

See here for security control mapping.