Topics for Networking Web Servers and Security

CDN

CDN (**C**ontent **D**elivery **N**etwork): servers that work together to **provide rapid delivery of Web content**. designed to minimize **latency** (network communication delays) for loading Web pages by **reducing the physical distance between the server and the user**. Also help protect against malicious attacks as **DDoS** (**D**istributed **D**enial **O**f **S**ervice). Enables quick migration of assets that needed to load content such as HTML pages, JavaScript, sheets, images, and videos.

***DDos: flood the network with malicious traffic.**

How CDN works

CDN (**C**ontent **D**elivery **N**etwork) stores a version of its contents in **cache** (high speed data storage layer) in several geographic locations called **POP** (**P**oint **O**f **P**resence). Each pop contains **cache servers** that designed to move content to visitors nearby.
Example: an IL visitor who would like to view content from a US based server would have to deal with slow loading times if the request had to travel across the Atlantic. To cancel the delay, CDN stores the content in a local pop at Israel.

CDN reduces loading times in the following ways:

- ✓ Allows users to connect to data center that is geographically closer - **Less distance means less time, plus it reduces load on the original server** (Instead of connecting to the location of the Web site's source server).
- ✓ **Load balancing** that helps data get to the user quickly.
- ✓ **Reduces amount of data transferred** by reducing file size by processes as file compression.
- ✓ Sites that use certificates, **TLS** (**T**ransport **L**ayer **S**ecurity) or **SSL** (**S**ecure **S**ockets **L**ayer) can be accelerated by CDN disc by streamlining re-use of TLS connections and running TLS.

☒ **The only way a user will know if they have access to CDN** (**C**ontent **D**elivery **N**etwork), **is if the given site address is different from the requested URL** (**U**niform **R**esource **L**ocator).

Caching

High-speed data storage layer. future requests for data, are submitted faster by accessing the primary storage location of the data.

How does caching work?

Increase data retrieval performance by reducing the need to access the underlying slow storage layer. Typically stored in a fast access hardware such as **RAM** (**R**andom **A**ccess **M**emory) and can also be used in software component correlation.

- ☒ **Typically stores a subset (index) of data temporarily, as opposed to databases that are usually full and durable.**

Latency

Network communications delays. The time it takes for data package to pass through several devices, get on the target, and decode it. The time it takes between the source and destination, measured in milliseconds.

- ☒ **Can affect at: satellite Internet connections, cable Internet connections, WiFi connections.**

WAF

WAF (**W**eb **A**pplication **F**irewall) protects Web applications from a variety of application layer attacks such as **cross-site scripting**, **SQL injection**, and **cookie poisoning**. Also help protect Web applications by filtering and tracking **HTTP** (**H**yper **T**ext **T**ransfer **P**rotocol) between app and Internet.

- ☒ **It operates through a set of rules called polices. Designed to protect against vulnerabilities in application by filtering malicious traffic.**

cross-site scripting (XXS): injection malicious scripts into trusted sites.

SQL injection: interrupt queries, attacker can view data that normally can't retrieve. can modify or delete data.

cookie poisoning: session hijacking. strategy of cookies to steal data, bypass security, or both.

CLOUD WAF

Advanced type of Firewall, designed for apps. Unlike a regular Firewall that stops communication to specific destinations, or from destinations, WAF filters specific content that is located within the communications.

WAF can update and match as needed. The media that passes through it, will be filtered by these rules and anything that does not meet them will be stopped. WAF can be modified, in addition to updates that enable it to adapt to new types of attacks (because various types of breaches are discovered all the time, and new attacks are quite advanced).

protect against attack types as:

Injection: Injecting hostile code into standard communications-driven information. In this way, even if the media itself is legitimate, the information it transfers will cause the damage the attacker planned.

Cross Site Scripting: Injecting scripts into different Web pages. A technique that can be used for assault in a variety of creative ways.

Insecure Cryptographic Storage: Situation that information is stored insecurely that can be exploited in a variety of malicious ways.

☒ **To achieve full protection against attacks, WAF is with built-in DDoS protection.**

WAF cloud is available over the Internet and has the benefits of **SaaS** (**S**oftware **a**s **a S**ervice): As customers, they pay only as per the use of the service, instead of paying a large sum at once for purchase the WAF software for local installation, or a hardware device with WAF software as well as the update and maintenance of the software under the warranty of the Service Provider.

WAF is also available as a hardware device (WAF Appliance), but cloud service is preferable in all aspects.

WAF service has unique advanced protection capabilities:

- Real-time monitoring and detection of malicious activity, or an exception to Web-based sites or applications.
- Smart system that learns and remembers regular usage patterns on a site or application, and blocks activities that exceed the same patterns.
- Full hold of all available attack types as defined by **OWASP** (**O**pen **W**eb **A**pplication **S**ecurity **P**roject): SQL Injection attacks for theft or destruction of database, Privilege escalation attacks by falsifying **Cookies** to breaking into a site without legal authorization, Cross Site Request Forgery attacks for information theft, blocking credit card numbers, and more.
- Zero-day web attacks that use current-discovered, non-defensive signatures are still being identified.
- Ability to block attacks and malicious automatic bots that dynamically change their IP addresses to avoid IP Blacklisting by digitally fingerprinting the computer they use (IP-agnostic Device Fingerprinting).
- **Automated Protection Policy** build capability optimized for new applications.
- Protection from **DDoS** attacks will be effective up to **1G** (wireless cellular technology first generation).
- Defense against attack on layers 3,4,7.

Imperva Web Application Firewall

Web application attacks prevent important transactions and steal sensitive data. Imperva **WAF** (**W**eb **A**pplication **F**irewall) analyzes traffic to your applications to stop attacks and ensure uninterrupted business operations.

Protect without noise

A noisy WAF forces to choose between blocking legitimate traffic or manually containing attacks your WAF let through. Imperva Research Labs ensure accuracy to WAF customers as the threat landscape changes.

Reduce web app risk

Automatic policy creation and fast rule propagation empower your security teams to use third-party code without risk while working at the pace of DevOps (**Dev**elopment and **Op**eration**s**).

Protect Web Applications and APIs

Imperva WAF is a key component of a **comprehensive WAAP** (**W**eb **A**pplication **a**nd **A**pi **P**rotection) stack that secures from edge to database, so the traffic you receive is only the traffic you want. Imperva provide the best website protection in the industry PCI (Payment Card Industry) compliant, automated security that integrates analytics to go beyond OWASP (Open Web Application Security Project), and reduces the risks created by third-party code.

API: Application Programming Interface

Imperva WAF secure:

- Active and legacy applications
- Third-party applications
- APIs & Microservices
- Cloud applications, containers, VMs and more

Deploy to fit your needs

SaaS (Software as a Service)**:** Adapt as quickly as your applications using automated policy creation and rule propagation. Minimize the workload for your team and let Imperva handle the policies and saving you time and money.

Appliance: Deploy WAF exactly where you need it – Physical or virtual appliance, combine with RASP (Runtime Application Self Protection). Decide how to best defend your applications using dynamic profiling and attack intelligence

Imperva WAF goes anywhere & works everywhere

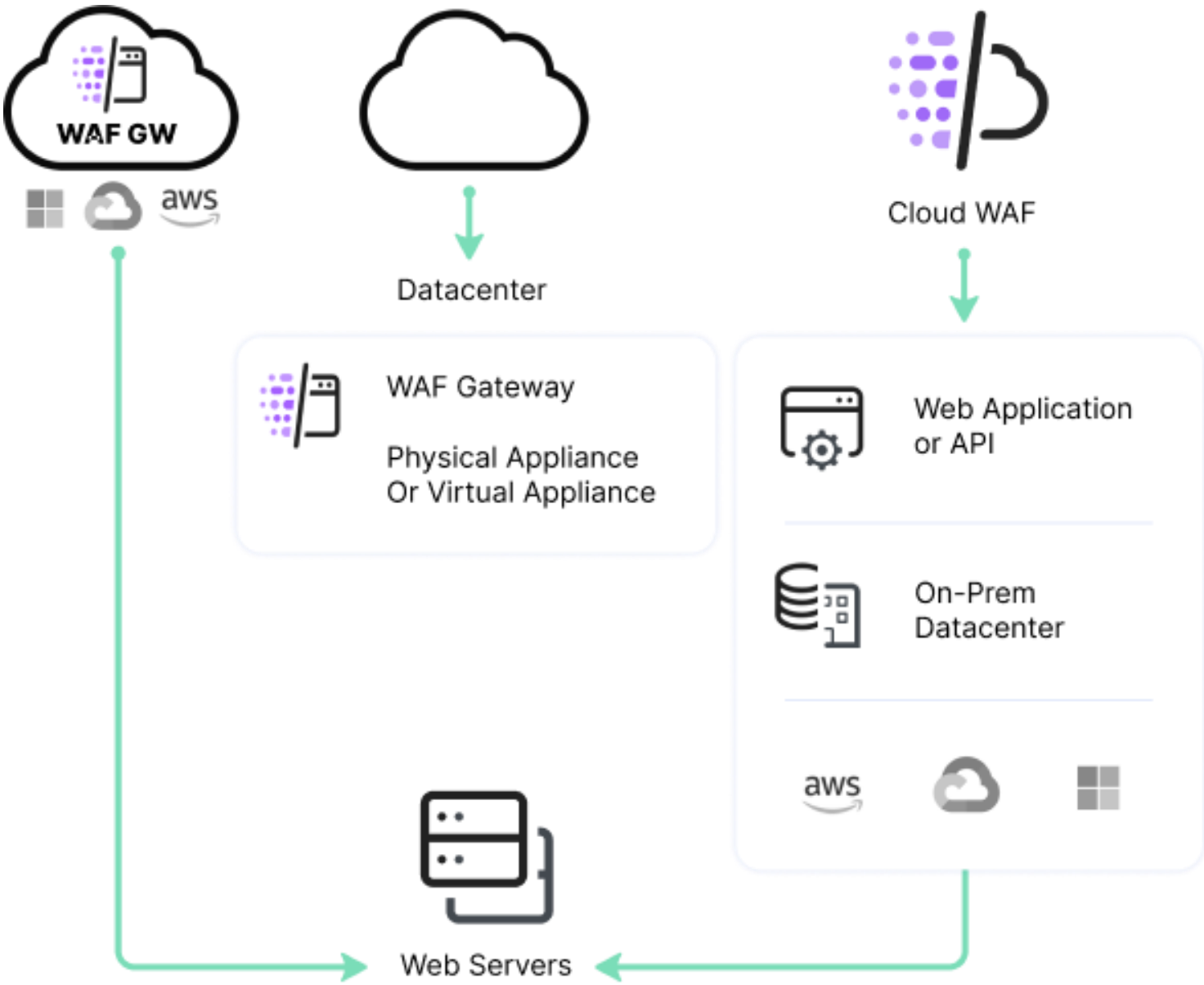
Advanced Security

Deploy Imperva WAF on-premises, in **AWS**, **Azure**, and **GCP**, or as a **cloud service**. Easily secure each application while meeting its specific service level requirement.

Imperva meet the unique needs by delivering the security solutions as a service or self-managed option. make sure you stay protected without disrupting your innovation delivery pipeline from modern threats including advance bots and API attacks.

Accurate & Customizable

Out-of-the-box rules for protection by default enable Imperva WAF's real-time technologies to close the loop on constantly changing attack patterns.



Centralize your configuration with a single stack approach, providing simplicity in provisioning, security and performance that go together to ensure better business continuity with fewer false positives.

NETWORKING

How does the Internet work?

Data moves over the Internet called **message**, but before sending messages, it divided into smaller parts called **packets** (contains source, data, destination. also known as **payload**).

- **Data (Messages** and **packets)** passed from one source to another by **IP** (**I**nternet **P**rotocol) and **TCP** (**T**ransport **C**ontrol **P**rotocol).

✓ **IP** and **TCP** work together to ensure that data transfer across the Internet is consistent and reliable, no matter which device users or where they use it.

IP (**I**nternet **P**rotocol) control how information sent from one computer to another by Internet connection. Using a numeric address (**IP address**) and the **IP system** is provided with additional instructions on how to transfer the data.

TCP (Traffic Control Protocol) works to ensure reliable data migration. It helps to make sure that no packets lost, the packets are re-assembled in a normal sequence, and there is no delay that affects the quality of the data.

OSI model

Open Systems Connection

Describe seven layers which computer systems use network communication.
The modern Internet is not based OSI, but on the simpler TCP/IP model. However, the OSI still widely used because It helps visualize and communicate how networks work and helping to isolate and solve network problems.

OSI 7 layers model

"top-down"

7	Application Layer	Human-computer interaction layer, where applications can access the network services
6	Presentation Layer	Ensures that data is in a usable format and is where data encryption occurs
5	Session Layer	Maintains connections and is responsible for controlling ports and sessions
4	Transport Layer	Transmits data using transmission protocols including TCP and UDP
3	Network Layer	Decides which physical path the data will take
2	Data Link Layer	Defines the format of data on the network
1	Physical Layer	Transmits raw bit stream over the physical medium

#	שם	בעברית	תפקיד	יישומים
7	Application	יישום	תקשורת עם המשתמש	דפדפן, YouTube, Skype, Facebook
6	Presentation	ייצוג	קידוד דחיסה וייצוג	CSS, HTML, GIF
5	Session	שיחה	אפשור קיום השיחה, בקרת דו-שיח, בקרת אסימון	HTTP, FTP
4	Transport	תעבורה	העברת נתונים בין שני הצדדים, אמינות	TCP, UDP
3	Network	רשת	העברת הנתונים ברשת מקצה לקצה - ניתוב	ICMP,RARP,IP, IPX, RIP
2	Data Link	קו	העברת הנתונים מנקודה לנקודה למרות הפרעות	Ethernet, Token ring,ARP
1	Physical	פיזית	העברת אותות, הגדרת מתחים, הגדרת חיבורים	RS232,802.11x WiFi, E1, 10Base-T, DSL

- בשלוש השכבות העליונות הפרוטוקולים השונים מוסיפים פתיח ו/או סוגר לנתונים, והם מכונים "רצף נתונים" (data stream). .
- בשכבת התעבורה הנתונים מחולקים למקטעים (segments) כדי שאפשר יהיה לבצע בקרה על ההעברה שלהם ברשת.
- בשכבת הרשת כל מקטע מחולק לחבילות (packets) ולכל חבילה מוצמדת הכתובת הלוגית של היעד.
- בשכבת הקו (data link) לכל חבילה מוצמדת הכתובת הפיזית של היעד, והחבילות מכונות מסגרות (frames).
- השכבה הפיזית עוסקת רק בייצוג הבינארי של המסגרות, ולכן מכונים - רצף בינארי (binary stream).

שכבה	מצב הנתונים	
Application	רצף נתונים	נתונים שונים, למשל טקסט כמו זה...
Presentation		
Session		
Transport	מקטעים	
Network	חבילות	
Data Link	מסגרות	
Physical	רצף בינארי	0110011011001100101...

7 Application layer: used by end-user software such as web browsers and e-mail clients.
It allows the software to send and receive information and present meaningful data to users.
protocols:
HTTP (**H**yper**T**ext **T**ransfer **P**rotocol), **FTP** (**F**ile **T**ransfer **P**rotocol), **POP** (**P**ost **O**ffice **P**rotocol), **SMTP** (**S**imple **M**ail **T**ransfer **P**rotocol), **DNS** (**D**omain **N**ame **S**ystem).

6 Presentation layer: Defines how two devices should encode, encrypt, and compress data so that it is received correctly on the other side. takes all the data transmitted by the application layer and prepares it to broadcast over the activation layer.

5 Activation layer: creates communication channels, called **sessions**, between devices. Responsible for opening sessions, ensuring that they remain open and functional while the data is being transferred, and closing when the communication is over.
can also set checkpoints during data migration - if activation is interrupted, devices can resume data transfer from the last checkpoint.

4 Transport layer: takes data that is transferred to the operating layer and breaks it into "**segments**" at the end of the transmitter. Responsible for re-assembling the sections at the end of the receiver and turn it back into data that can be used by the activation layer. performs flow control, sends data at a rate that matches the receiving device's connection speed and error control, checks to see if data has been received incorrectly and if not, requests it again.

3 Network layer (two main functions):

- **Breaks apart segments** into network packages and re-assembles the packets at the receiving end.
- **Routing packets** by discovering the best path on a physical network. The network layer uses network addresses, usually **IPA** (**I**nternet **P**rotocol **A**ddresses) to route packets to a target node.

2 Data link layer - creates and stops connection between two nodes that physically connected to the network. Breaks rations into frames and sends them from source to destination.
consists two parts:

- **LLC** (**L**ogical **L**ink **C**ontrol) - identifies network protocols, performs error checking, and synchronizes frames.
- **MAC** (**M**edium **A**ccess **C**ontrol) - control media that uses MAC addresses to connect devices and sets permissions to transmit and receive data.

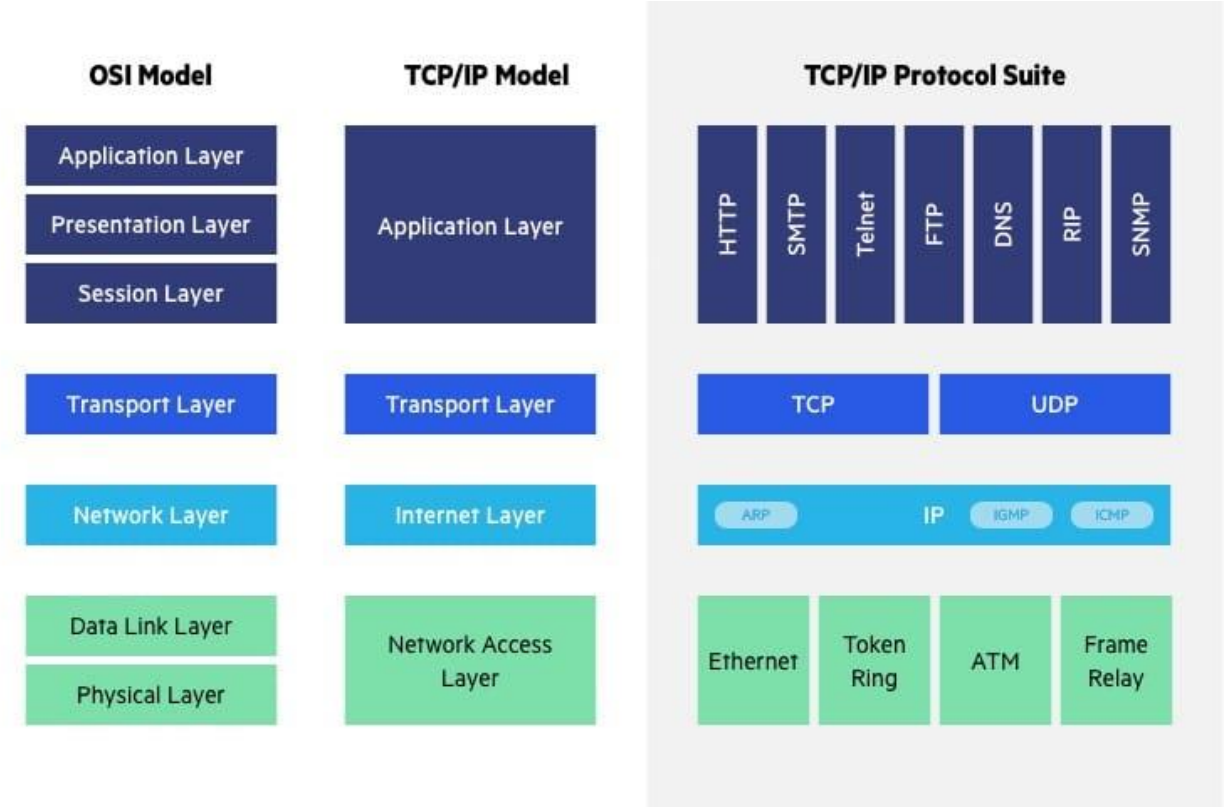
1. Physical layer: physical cable or wireless connection between connections in the network.
It defines the **connector, power cable**, or **wireless technology** that connects the devices, and responsible for the transmission of raw data, which is simply a series of **binary** 0 and 1, while maintaining bitrate control.

The benefits of the OSI model

Helps **users** and **operators**:

- Set the hardware and software that required to build their network.
- Understand and communicate the process, followed by components that communicate over the network.
- Troubleshoot problems by identifying which network layer causes a problem and focusing on this layer.

OSI versus TCP/IP model



The main difference that **TCP/IP** (**T**he **C**ontrol **P**rotocol/**I**nternet **P**rotocol) are simpler, and several **OSI** layers get into one:

- OSI **5, 6, 7** layers are integrated into one application layer at tcp\ip.
- OSI **1, 2** layers are integrated into one network access layer at tcp\ip.

Other important differences:

- **TCP/IP** designed to **solve specific forms of communication problems**.
- **OSI** designed to **describe all forms of communication on the network**.
- In TCP/IP, most applications use all layers, while OSI simple applications don't use all seven layers. Only layers **1, 2, 3** are required to allow any data communication.
- TCP/IP does not take responsibility for sequence and approval functions.

Imperva's security solutions

Secure applications across multiple layers of OSI from the network layer:

- **DDoS**
  Distributed **D**enial **O**f **S**ervice

- **WAF**
  Web **A**pplication **F**irewall

- **bot management**
  blocking undesired or malicious Internet bot traffic while still allowing useful bots to access web properties

- **API**
  Application **P**rogramming **I**nterface

DNS

DNS (**D**omain **N**ame **S**ystem) translate domain, as **google.com** or **Ynet.co.il** into IP (**I**nternet **P**rotocol) addresses and control which end user server will arrive when they type domain name in their web browser. These requests are called queries.
Each Internet connected device has a unique IP address that other machines use to find the device.
DNS servers eliminate the need for humans to memorize IP addresses (such as IPv4 8.8.8.8, or newer alphanumeric IP addresses as IPv6 2001:4860:4860::8888).

There are 4 DNS servers involved in loading a Web page:

1. **DNS recursor (recursive resolver)** - Receive queries from client using applications such as Web browsers. responsible for making more requests to provide the customer's DNS query (**request** for information sent from a **DNS** Client to a **DNS** Server. When **DNS** Client needs to find the IP Address of computer known by its **FQDN** (**F**ully **Q**ualified **D**omain **N**ame), it **queries DNS** servers to get the **IP Address**).

2. **Root name server** - First stage in translation host names that are readable for IP addresses.

3. **TLD** (**T**op **L**evel **D**omain) Name server - next step in searching for a specific IP address, it hosts the last part of the host name (example: **.com**, the **TLD** server is "**com**").

4. **Authorized name server** - Last stop on the name server query. If the server has access to the requested record, it will return the IP address of the requested host name to DNS that submitted the initial request.

Diagnostics tools

- **MTR** (**M**y **T**race**R**oute) - **combines traceroute and ping**. common method for testing connectivity and network speed. MTR constantly displays updated information about the delay, and packet loss along the route to the destination.

- **Ping** (**P**acket **I**nter **N**etwork **G**roper) - Used to test if a particular host can be reached on an IP network. Measures the time it takes to send a packet from the target computer to the destination computer and back. also measures and records the back times and all loss of information along the way.

SSL

SSL (**S**ecure **S**ockets **L**ayer) - Secure connection between two or more devices over the Internet.
encryption system that uses two keys to encrypt: public key that everyone knows,
and a private key that known only to the recipient of the message.
many sites use the protocol to obtain confidential user information including credit card numbers.
Web addresses that require an SSL connection begin with https instead of http.

☒ **Both SSL and HTTPS have been approved as standards.**
- **SSL is secure connection between client and server.**
- **HTTPS deliver individual messages securely.**

How does SSL work?

SSL runs by a **three-step handshake application** that composed over **TCP** (**T**raffic **C**ontrol **P**rotocol) connection. This allows to share encrypted data between the browser and the server, as is associated with a https label instead of http. When a Web browser tries to connect to a site using SSL, the browser first prompts the Web server to identify itself. This prompts the Web server to send a copy of the SSL certificate to the browser. The browser checks whether SSL is trusted and if so, the browser sends an authentication message to the Web server. The server then responds to a browser with a digitally signed certificate to run an encrypted SSL session.

TLS

TLS (**T**ransport **L**ayer **S**ecurity) - ensures privacy and integrity of data between client and server applications that communicate over the Internet.

Consists of two layers:

☒ **TLS Record** - such as TCP (**T**raffic **C**ontrol **P**rotocol), its ensures that connection is private with symmetric data, encryption and ensures that the connection is reliable. also used to seal higher-level protocols, as TLS Handshake.

☒ **TLS Handshake** - allows authentication between the server and client and the negotiation of an encryption algorithm, and cryptographic keys **before the application protocol transmits or receives any data**.

☒ **TLS doesn't depend on application protocol. Higher-level protocols can integrate over TLS in a transparent way.**

TSL vs SSL

TLS (**T**ransport **L**ayer **S**ecurity) - refiners the SSL handshake process and improves some of the security vulnerability to create more reliable protocol. TSL credentials sometimes referred as SSL certificates.

SSL (**S**ecure **S**ockets **L**ayer) is older than TLS. Considered vulnerable due **POODLE attacks** (**P**adding **O**racle **O**n **D**owngraded **L**egacy **E**ncryption) which allowed to **steal secure HTTP cookies**, **HTTP permission**, or **HTTP content**. Today **SSL** is still widely deployed.

HTTP/HTTPS

- **HTTP** (**H**yper**T**ext **T**ransfer **P**rotocol) - gives users a way to communicate with Web resources as HTML files by forwarding hypertext messages between clients and servers.
  HTTP typically use **TCP** (**T**raffic **C**ontrol **P**rotocol) connections to communicate with servers and acts as a request-response protocol between client and server.

HTTP request and response structure

Request:

1. In the request line bar place the HTTP method to use.
2. The request **URI** (**U**niform **R**esource **I**dentifier) and the HTTP protocol to use.
3. Request Title.
4. Body request.

Response:

After accepting and requiring request message, the server responds with HTTP response message:

1. Status bar.
2. Zero or more header fields (General | Response | Entity) followed by CRLF (special character elements). These components are embedded in HTTP header sand other software code to highlight AOL (End of Line).
3. An empty row (a row with nothing ahead of CRLF) that indicates the end of the heading fields.

Example: Client (browser) sends an HTTP request to the server; Then the server returns a response to the client. The response contains status information about the request and may also contain the requested content.

HTTP Status Codes

HTTP shares the status of a request with the requester using status codes, which are sent back along with the response. There are five classes of status codes:

1. Informational (1XX)
2. Successful (2XX)
3. Redirection (3XX)
4. Client Error (4XX)
5. Server Error (5XX)

Example: 100 ('Continue') is reported during a connection with the client and indicate that the server is continuing to process the request. 200 ('OK') indicate that a complete response has been correctly prepared.

Seven HTTP methods are:

1. Get: "read-only" request for information from a server.
2. Post: Triggered from forms in browsers when the form method is set to POST.
3. Head: identical to the GET method, but doesn't return a message body
4. OPTIONS: Tell the requester what HTTP methods can be executed against the URL requested.
5. Trace: The purpose of TRACE is to return the request back to the requester in the state it was in at the point where it reached the last computer in the chain.
6. Put: Take a client resource and put it in a location on a server as specified in the URL of the request.
7. Delete (or CRUD - **C**reate**, R**ead**, U**pdate**, D**elete): causes the server to delete the contents of the target URL.

- **HTTPS** (**H**yper**T**ext **T**ransfer **P**rotocol **S**ecure) - uses the communication encryption protocol.
  This protocol secures communication using a **public asymmetric infrastructure key**. This key is used to decrypt information encrypted by the **public key** (for those who want to securely interact with the server). The protocol was called **TLS** (**T**ransport **L**ayer **S**ecurity) and known as **SSL** (**S**ecure **S**ockets **L**ayer).

Structure

SSL (**S**ecure **S**ockets **L**ayer) and **TLS** (**T**ransport **L**ayer **S**ecurity) refer to the type of secure connection that uses HTTPS in communication between Web browser and Web server.
Step 1: Purchase of SSL/TLS certificate.
Step 2: Install SSL certificate on the site.
Step 3: Change site settings to use HTTPS.

WEB SERVERS

The primary role of Web server is to host sites, or store data for Web-based sites and applications or Web applications. Web servers also support **SMTP** (**S**imple **M**ail **T**ransfer **P**rotocol) and **FTP** (**F**ile **T**ransfer **P**rotocol), which used for **e-mail**, **file transfer**, and **storage**.

- **Web server hardware** connected to the Internet and allowed data to be exchanged with other connected devices.
- **Web server software** controls how user accesses the hosted files

(All computers that host sites must have Web server software).

How do Web servers work?

Web server software accesses the domain names of sites and ensures that the site content is delivered to the requested user. As a hardware, a Web server is a computer that stores Web server software and other site-related files, as **documents**, **HTML images**, and **JavaScript files** and Can also host several domains on one Web server.

When Web browser, such Google Chrome or Firefox, needs a file hosted on a Web server, the browser will request the file using HTTP. when the request received by the Web server, the HTTP server will accept the request, find the content, and send it back to the browser using HTTP, then the browser can view the web page. If the requested page does not exist, or if something goes wrong, the Web server will respond with an error message.

More precisely- when a browser requests a page from a Web server, the process will perform a series of steps: First, person will specify a URL in the web browser address bar. The web browser will then accept the domain name IP address, (or translate the site address using DNS, or by searching its cache). This will bring the browser to the web server. Then the browser prompts the specific file from the Web server using the HTTP request that the Web server sends to the browser the requested page again using HTTP.

Examples of Web server usage

- Send and receive emails.
- Download requests for protocol and **FTP** (**F**ile **T**ransfer **P**rotocol).
- Build and publish Web pages.
- Support server-side scripting, which used to deliver scripts on Web server that can customize the response to the client.

Server-side scripts run on the server computer and typically have a wide feature set that includes access to database. server-side scripting also uses **ASP** (**A**ctive **S**erver **P**rotocol), **PHP** (originally **P**ersonal **H**ome **P**age**, now HyperText Preprocessor**), and other scripting languages. This process also enables to dynamically create HTML documents.

static vs Dynamic Web servers

- **Static** - Content that displayed as is. Considered a static because the router will send guest files like a browser.
- **Dynamic** Content that can update and change. Application server can be used to update all files that hosted before they sent to the browser. This process is more flexible, but it's also more complicated.

common web servers:

- **Apache HTTP Server:** free open-source web server for Windows, Mac OS X, Unix, Linux, Solaris, and other operating systems - needs the Apache license.
- **IIS** (**I**nternet **I**nformation **S**ervices)**:** Developed by Microsoft for Microsoft platforms. not open sourced, but widely used.
- **Nginx:** popular open-source web server for administrators because its light resource utilization and scalability. It can handle many concurrent sessions due to its event-driven architecture. Nginx also can be used as a proxy server and load balancer.
- **Lighttpd:** free web server that comes with the FreeBSD operating system. It is seen as fast and secure, while consuming less CPU power.
- **Sun Java System Web Server:** free web server from Sun Microsystems that can run on Windows, Linux, and Unix. It is well-equipped to handle medium to large websites.

☒ **Choosing web server include how well it works with operating system and other servers, His ability to handle server-side programming, Security Properties, advertising, search engine and tools for building sites. The Web servers may also have different configurations and set default values.**

Examples of Web server security methods

- **Reverse proxy** designed to hide an internal server and serve as a traffic broker from an internal server
- **Restrict Web host access** to infrastructure machines or use **Secure SSH** (**S**ocket **Sh**ell)
- Maintaining an updated, revised Web server to ensure that the Web server is not vulnerable
- Network monitoring to ensure no activity or unauthorized
- **FW and SSL** as firewall to monitor **HTTP traffic** while having an **SSL** (**S**ecure **S**ockets **L**ayer) can help keep data secure.

SECURITY

Applications and networks

Command Cyber attack

XSS (**C**ross **S**ite **S**cripting): Form of injection, which malicious scripts are injected into trusted sites. Occur when an attacker uses application to send malicious code to another end user. Usually in the form of a script on the side of the browser.
SQLI (**SQL I**njection): allow attacker to interrupt queries the application is performing to its database. Allows an attacker to view data that they normally can't retrieve. This may include data that belongs to other users, or any other information that the application itself can access. Attacker can modify or delete this data, causing ongoing changes to the content or behavior of the app. Attacker could compromise the underlying server, or perform a denial-of-service attack.
MITM (**M**an **I**n **T**he **M**iddle): When attacker places himself in conversation between user and app (pose as one of the parties), making it appear as if there is a regular exchange of information.
DDOS (**D**istributed **D**enial **O**f **S**ervice): Flood network with so much malicious traffic that it cannot operate or communicate as normally does.

Imperva application security solution includes:

- **DDoS protection:** maintains operational time in all situations. Prevents any kind of DDoS attack of any size, prevent access to the site and network infrastructure.
- **CDN**: improves site performance and reduces bandwidth costs with a CDN designed for developers. Static resource cache at end while accelerating API interfaces and dynamic sites.
- **WAF**: a cloud-based solution that permits traffic and eliminates legitimate malicious traffic, application retention at the edge. **Gateway WAF** keeps applications and APIs within the network safe.
- **Bot protection:** analyzes bot traffic for anomalies, detects poor bot behavior, and is authenticated by mechanisms that do not affect user traffic.
- **API security:** protects APIs by ensuring that desired traffic can access the API endpoint as well as detect and block vulnerabilities.
- **account takeover Protecting:** uses a intent-based identification process to identify and protect against attempts to take over user accounts for malicious purposes.
- **RASP** (**R**untime **A**pplication **S**elf **P**rotection): Keep the applications on the inside from known attacks. Fast, accurate protection without signature or learning status.
- **attack Analytics:** mitigate and respond to real cyber security threats efficiently and accurately with actionable intelligence across all layers of defense.

DDoS Protection

Imperva DDoS Protection secures all assets at the edge for un-interrupted operation. Ensure business continuity with guaranteed uptime.

Minimize downtime

When it comes to DDoS mitigation, the rule of thumb is:
"moments to go down, hours to recover".
Therefore, when defending against an attack, every second counts. Imperva ensures business continuity, with guaranteed uptime, and no performance impact.

Avoid bandwidth costs

Imperva gives you the peace of mind that attack traffic will be automatically blocked at the edge – without you having to scale up in bandwidth to pay for it.
Unlimited protection against attacks of any size or duration.

DDoS Protection service:

Maximum Visibility

- ✓ **Instant attack notifications** Mail, SMS, and mobile app
- ✓ **Easy monitoring** Network traffic and application analytics
- ✓ **Uncover the real threats** Layer 3,4 and Layer 7 event correlation via Attack analytics
- ✓ **Integration with your SIEM** Works seamlessly with the leading SIEMs

Optimized Performance

- ✓ **Quick deployment and scaling** Software-defined network (SDN) automated tuning
- ✓ **The best traffic routing** Advanced Anycast at the edge
- ✓ **Real-time capacity management** Load conditions, granularity of Internet pipes for every point of presence (PoP)
- ✓ **Minimal latency to 95% of the globe** Global mesh network

Detecting Denial-of-Service and Distributed Denial-of-Service Attacks

Denial-of-service (DoS) attacks can take many forms, but the goal remains the same: preventing access to a system or service. They can be conducted from a single system, or from many systems as part of a distributed denial-of-service (DDoS) attack. Detecting and preventing DoS attacks is an increasingly important part of a cybersecurity analyst's skillset.

DoS Attacks

DoS attacks typically include one or more of the following patterns of attack:

- Attempts to overwhelm a network or service through the sheer volume of requests or traffic
- Attacks on a specific service or system vulnerability to cause the system or service to fail
- Attacks on an intermediary system or network to prevent traffic from making it between two locations

Each of these types of attacks requires slightly different methods of detection. This means that your network, system, and service monitoring capabilities need to be set up to monitor for multiple types of attacks depending on which might target your infrastructure.

A DoS attack from a single system or network can typically be stopped by blocking that system or network using a firewall or other network security device. IPSs can also block known attack traffic, preventing a DoS attack from occurring. Single-system DoS attacks are not as likely as DDoS attacks unless the target suffers from a specific service or application vulnerability, or the target can be easily overwhelmed by a single remote system due to limited bandwidth or other resources.

Distributed Denial-of-Service Attacks

Distributed denial-of-service (DDoS) attacks come from many systems or networks at the same time. They can be harder to detect due to the traffic coming from many places, and that also makes them much harder to stop. Many DDoS attacks are composed of compromised systems in botnets, allowing attackers to send traffic from hundreds or thousands of systems.

Tools like the Low Orbit Ion Cannon (LOIC) have also made participation in DDoS attacks a voluntary effort as part of hacktivist efforts from groups like Anonymous. Understanding why your organization might be targeted, and by whom, is an important part of planning for and responding to DoS and DDOS attacks.

Detecting DoS and DDoS Attacks

Since there are many flavors of DoS and DDoS attacks, building an effective DoS and DDoS detection capability usually involves multiple types of tools and monitoring systems. These often include the following:

- Performance monitoring using service performance monitoring tools
- Connection monitoring using local system or application logs
- Network bandwidth or system bandwidth monitoring
- Dedicated tools like IDS or IPSs with DoS and DDoS detection rules enabled

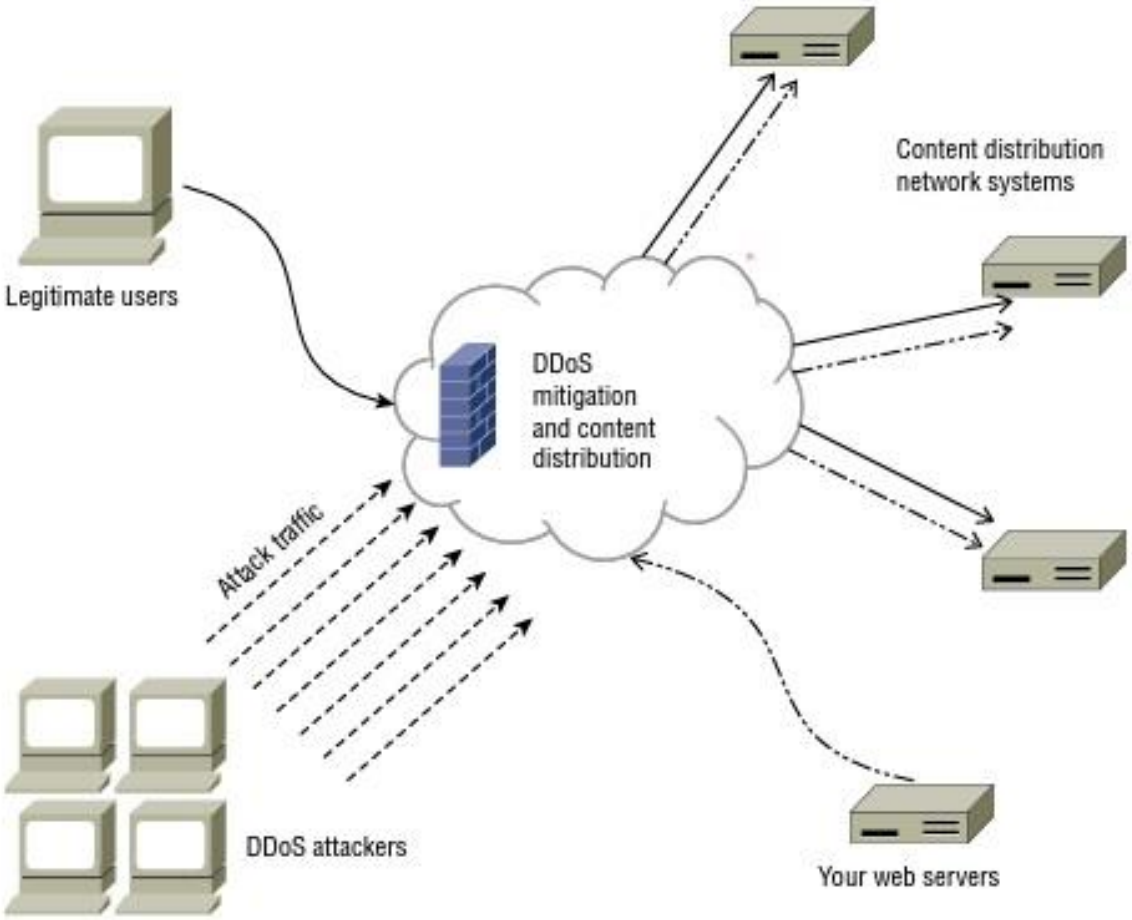
During incident response, the same command-line tools that you can use to analyze network traffic (like netstat) can help with troubleshooting on local servers, but a view from the network or service perspective will typically provide a broader view of the issue.

Surviving a Denial-of-Service Attack

If your organization has a public Internet presence, you're likely to have to deal with a DoS attack at some point, whether it's on purpose or accidental. Fortunately, services and tools now exist to help organizations weather these attacks.

Here are two common ways to survive a DDoS attack:

- Using a dedicated service designed to handle these attacks that uses a large distributed network of endpoints combined with DDoS mitigation tools to ensure that your service (typically a website) can be accessed even if one or more distribution locations is offline. In the following graphic, a DDoS mitigation system distributes copies of a website's content to globally distributed content distribution network (CDN) servers while blocking DDoS attacks using a centrally managed defense mechanism. This ensures that legitimate users receive a response from a CDN node that is close to them, avoiding potential issues with the main website or networks that serve it during a DDoS.



- Deploying a DDoS mitigation device or technology. These often analyze flows or sit in-line between the protected systems and the public Internet. They then gather data to provide a view of traffic to a network or service and redirect or drop bad traffic based on signatures or behavior analysis.

API (**A**pplication **P**rogramming **I**nterface)

ASP (**A**ctive **S**erver **P**rotocol)

CDN (**C**ontent **D**elivery **N**etwork)

DDOS (**D**istributed **D**enial **O**f **S**ervice)

DAF (**D**ns **A**pplication **F**irewall)

DNS (**D**omain **N**ame **S**ystem)

FTP (**F**ile **T**ransfer **P**rotocol)

HTTP (**H**yper **T**ext **T**ransfer **P**rotocol)

HTTPS (**H**ypertext **T**ransfer **P**rotocol **S**ecure)

IP (**I**nternet **P**rotocol) and **IP system**

IPA (**I**nternet **P**rotocol **A**ddresses)

IPS (**I**ntrusion **P**revention **S**ystem)

LAN (**L**ocal **A**rea **N**etwork)

Latency (Network communications delays)

LLC (**L**ogical **L**ink **C**ontrol)

NAT (**N**etwork **A**ddress **T**ranslation)

NAC **(N**etwork **A**ccess **C**ontrol)

MAC (**M**edium **A**ccess **C**ontrol)

MITM (**M**an **I**n **T**he **M**iddle)

MTR **(M**y **T**race**R**oute**)**

OSI (**O**pen **S**ystems **C**onnection **M**odel)

OWASP (**O**pen **W**eb **A**pplication **S**ecurity **P**roject)

packets (source, data, destination**)**

PAT (**P**ort **A**ddress **T**ranslation)

PHP (originally **P**ersonal **H**ome **P**age now **H**ypertext **P**reprocessor)

Ping (**P**acket **I**nter **N**etwork **G**roper)

POODLE (**P**adding **O**racle **O**n **D**owngraded **L**egacy **E**ncryption)

POP (**P**ost **O**ffice **P**rotocol)

RAM (**R**andom **A**ccess **M**emory)

SMTP (**S**imple **M**ail **T**ransfer **P**rotocol)

SQLI (**SQL I**njection)

SSL (**S**ecure **S**ockets **L**ayer)

SSH (**S**ecure **Sh**ell or **S**ecure **S**ocket **Sh**ell)

TCP (**T**raffic **C**ontrol **P**rotocol)

TCP/IP (**T**raffic **C**ontrol **P**rotocol/**I**nternet **P**rotocol)

TLD **(T**op **L**evel **D**omain) Name server

TLS (**T**ransport **L**ayer **S**ecurity)

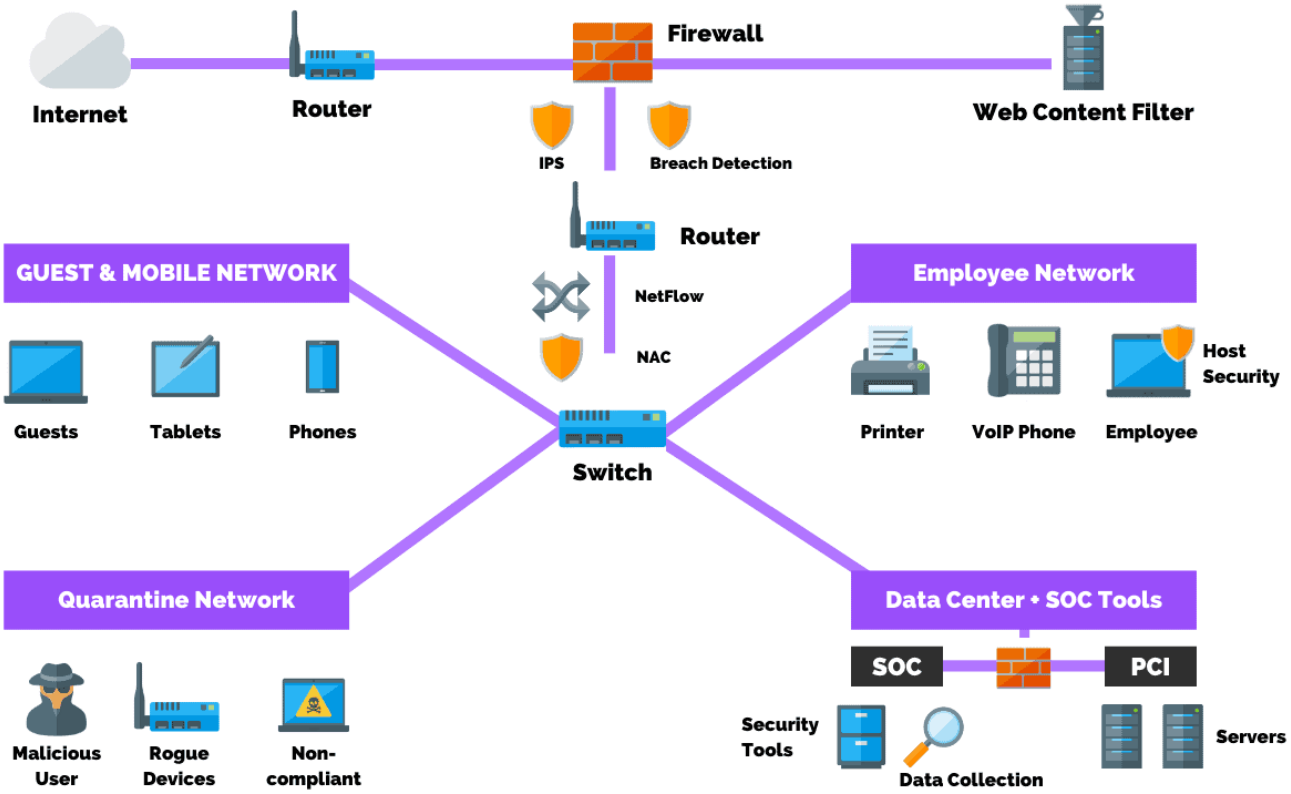
URI (**U**niform **R**esource **I**dentifier)

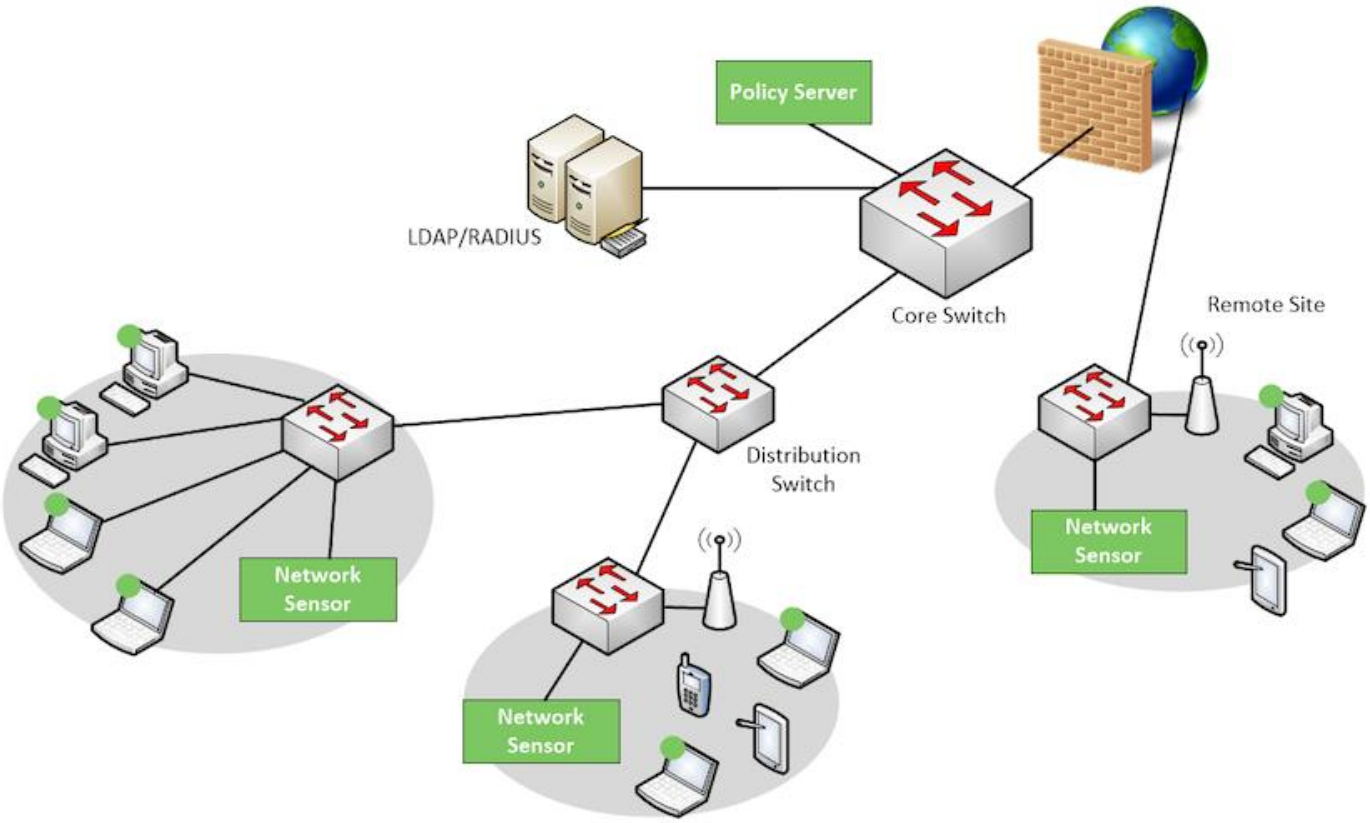
URL (**U**niform **R**esource **L**ocator)

WAN (**W**ide **A**rea **N**etwork)

WAF (**W**eb **A**pplication **F**irewall)

XSS (**C**ross **S**ite **S**cripting)


Internet
Router
Firewall
Web Content Filter
IPS
Breach Detection
Router
NetFlow
NAC
GUEST & MOBILE NETWORK
Guests
Tablets
Phones
Employee Network
Printer
VoIP Phone
Employee
Host Security
Switch
Quarantine Network
Malicious User
Rogue Devices
Non-compliant
Data Center + SOC Tools
SOC
PCI
Security Tools
Data Collection
Servers


Policy Server
LDAP/RADIUS
Core Switch
Remote Site
Distribution Switch
Network Sensor
Network Sensor
Network Sensor