## **Security Requirements:**

Software owners don’t just accept any new software features that are deployed.The features must go through a strategic process of critique, justification, and analysis before being deployed. The teams should treat security with the same attention to detail. After all, secure software doesn’t just happen out of nowhere it has to be a requirement of the strategic development process. To deploy secure software effectively, we need clear, consistent, testable, and measurable software security requirements.

### ● **Secrets Management**

Secrets are often mundane. Even basic static websites need to keep secrets. For most applications, secrets take the form of database passwords and encryption private keys. Even for static sites, secrets can take the form of keys used to connect to deployment systems or passwords for the web hosting server. As a developer, it’s your responsibility to protect these secrets. If someone malicious gained access to your application secrets, they’d gain access to whatever those secrets protect. Like we already mentioned, this might mean leaking your database data. If the secret is instead the password to a hosting server, they could replace your website with malware or deface it to spread a message you don’t support.

### ● **Application Firewalls**

A WAF protects your web apps by filtering, monitoring, and blocking any malicious HTTP/S traffic traveling to the web application, and prevents any unauthorized data from leaving the app. It does this by adhering to a set of policies that help determine what traffic is malicious and what traffic is safe. Just as a proxy server acts as an intermediary to protect the identity of a client, a WAF operates in similar fashion but in the reverse called a reverse proxy acting as an intermediary that protects the web app server from a potentially malicious client.WAFs can come in the form of software, an appliance, or delivered as-a-service. Policies can be customized to meet the unique needs of your web application or set of web applications. Although many WAFs require you update the policies regularly to address new vulnerabilities, advances in machine learning enable some WAFs to update automatically. This automation is becoming more critical as the threat landscape continues to grow in complexity and ambiguity.

### ● **SAST**

Static application security testing (SAST), or static analysis, is a testing methodology that analyzes source code to find security vulnerabilities that make your organization’s applications susceptible to attack. SAST scans an application before the code is compiled. It’s also known as white box testing.

SAST takes place very early in the software development life cycle (SDLC) as it does not require a working application and can take place without code being executed. It helps developers identify vulnerabilities in the initial stages of development and quickly resolve issues without breaking builds or passing on vulnerabilities to the final release of the application.SAST tools give developers real-time feedback as they code, helping them fix issues before they pass the code to the next phase of the SDLC. This prevents security-related issues from being considered an afterthought. SAST tools also provide graphical representations of the issues found, from source to sink. These help you navigate the code easier. Some tools point out the exact location of vulnerabilities and highlight the risky code. Tools can also provide in-depth guidance on how to fix issues and the best place in the code to fix them, without requiring deep security domain expertise.

### ● **In-built security**

Security measures include improving security practices in the SDLC and throughout the application lifecycle. All appsec activities should minimize the likelihood that malicious actors can gain unauthorized access to systems, applications or data. The ultimate goal of application security is to prevent attackers from accessing, modifying or deleting sensitive or Proprietry data.Any action taken to ensure application security is a counter measure or security control. The National Institute of Standards and Technology defines a security control as: "A safeguard or countermeasure prescribed for an information system or an organization designed to protect the confidentiality, integrity, and availability of its information and to meet a set of defined security requirements."

### ● **Security standard compliances**

An information security compliance management program comprises a minimum set of security requirements for protecting data that apply to any organization that stores, processes, or transmits that data.Compliance as Code is the process of using automated tools to review code so that teams can build compliance into development and operations. By incorporating compliance policies, checks, and auditing into development, regulatory compliance is no longer a time-consuming burden that development teams need to overcome.Although subtly different in wording and emphasis, the standards across industries follow a similar approach to ensuring the development of safe and/or secure systems. This common approach includes ten phases:

1. Perform a system safety or security assessment.
2. Determine a target system failure rate.
3. Use the system target failure rate to determine the appropriate level of development rigor.
4. Use a formal requirements capture process
5. Create software that adheres to an appropriate coding standard
6. Trace all code back to their source requirements
7. Develop all software and system test cases based on requirements
8. Trace test cases to requirements
9. Use coverage analysis to assess test completeness against both requirements and code
10. For certification, collect and collate the process artifacts required to demonstrate that an appropriate level of rigor has been maintained.

## **Reliability Requirements:**

### ● **Auto Scaling**

Autoscaling is a cloud computing feature that enables organizations to scale cloud services such as server capacities or virtual machines up or down automatically, based on defined situations such as traffic ir utilization levels. Cloud computing providers, such as Amazon Web Services, Microsoft Azure, and Google Cloud Platform (GCP), offer autoscaling tools.Core autoscaling features also allow lower cost, reliable performance by seamlessly increasing and decreasing new instances as demand spikes and drops. As such, autoscaling provides consistency despite the dynamic and, at times, unpredictable demand for applications.The overall benefit of autoscaling is that it eliminates the need to respond manually in real-time to traffic spikes that merit new resources and instances by automatically changing the active number of servers. Each of these servers requires configuration, monitoring and decommissioning, which is the core of autoscaling.

**Auto scaling provides several advantages:**

Cost. When loads are low, auto scaling allows both companies managing their own infrastructure and businesses that rely on cloud infrastructure to send some servers to sleep. This reduces electricity costs and water costs where water is used in cooling. Cloud auto scaling also means paying for total usage instead of maximum capacity.

Security. Auto scaling also protects against application, hardware, and network failures by detecting and replacing unhealthy instances while still providing application resiliency and availability.

Availability. Auto scaling improves availability and uptime, especially when production workloads are less predictable.

While many businesses have a set daily, weekly, or yearly cycle to govern server use, auto scaling is different in that it reduces the chance of having too many or too few servers for the actual traffic load. This is because auto scaling is responsive to actual usage patterns, in contrast to a static scaling solution.

### ● **Load Balancing**

Load balancing is the process of redistribution of workload in a distributed system like cloud computing ensuring no computing machine is overloaded, under-loaded or idle . Load balancing tries to speed up different constrained parameters like response time, execution time, system stability etc.

There are two load balancing methods:

* Round robin: Round robin load balancing distributes traffic to a list of servers in rotation using the Domain Name System (DNS). ...
* Weighted round robin: Allows an administrator to assign different weights to each server.

**Google Cloud offers the following load-balancing features:**

* Single anycast IP address.
* Software-defined load balancing.
* Seamless autoscaling.
* Layer 4 and Layer 7 load balancing.
* External and internal load balancing.
* Global and regional load balancing.
* Advanced feature support.

Elastic Load Balancing and application autoscaling are closely related. Both application auto scaling and load balancing reduce backend tasks such as monitoring the health of servers, managing the traffic load among the servers, and increasing or reducing servers pursuant to requirements. In fact, it is common to see solutions that include a load balancer with autoscaling features. However, elastic load balancing and auto scaling are distinct concepts.

### ● **Cloud based and Server less deployments**

Serverless is a cloud-based code execution model where cloud providers, instead of developers, deal with servers and computing resource management. There are no virtual machines or physical servers: they are deployed automatically in the cloud by vendors.Cloud providers take care of provisioning, maintaining, and scaling the serverless architecture. What’s more, the serverless architecture allows launching apps as needed: you don’t pay for ‘always-on’ server components to run your app when it’s not being used. Instead, some event triggers app code, and the resources are dynamically allocated for that code. You stop paying as soon as the code is executed.So, in a nutshell, serverless architecture is a way to build your cloud based application without managing infrastructure. It eliminates the need for routine tasks like security patches, capacity management, load balancing, scaling, etc. Still, serverless does not mean there are no servers at all. The term is somewhat elusive. Servers are simply eliminated from the app development since they are managed by the vendors.

### ● **Fault tolerance**

Cloud fault tolerance simply means your infrastructure is capable of supporting uninterrupted functionality of your applications despite failures of components.In a cloud computing setting that may be due to autoscaling across geographic zones or in the same data centers. There is likely more than one way to achieve fault tolerant applications in the cloud in most cases. The overall system will still demand monitoring of available resources and potential failures, as with any fault tolerance in distributed systems.Fault tolerance in cloud computing is about designing a blueprint for continuing the ongoing work whenever a few parts are down or unavailable. This helps the enterprises to evaluate their infrastructure needs and requirements, and provide services when the associated devices are unavailable due to some cause.

### ● **Recoverability**

Recoverability requirements that describe the ability to recover from failed states and return the system to its as-built condition. Using the example of a failed unit of hardware, a resilient system will continue to function after failure, while a recoverable system will have a simple and predictable method for recovering from the hardware failure. Data backups, data replication, hot-swap hard drives, and automated operating system and application deployment tools may be technologies or techniques to recover a failed component. There are four Recoverability requirements: Component Recovery, Site Recoverability, Configuration Recovery and Logical Recovery. Each requirement may have multiple levels with each metric.

Following steps should be performed to obtain optimum recovery testing performance:

1. Recovery Analysis. Before starting the recovery process it is important to analyze the system's ability to recover. ...
2. Preparation of test plan. ...
3. Preparation of test environment. ...
4. Maintaining Backup.

In Software Engineering, Recoverability Testing is a type of Non-Functional Testing. (Non- functional testing refers to aspects of the software that may not be related to a specific function or user action such as scalability or security.)

The time taken to recover depends upon:

* The number of restart points
* A volume of the applications
* Training and skills of people conducting recovery activities and tools available for recovery.