Desmistificando Microsserviços e DevOps: Projetando Arquiteturas Efetivamente Escaláveis

Prof. Vinicius Cardoso Garcia
vcg@cin.ufpe.br :: @vinicius3w :: assertlab.com

[IF1004] - Seminários em SI 3
https://github.com/vinicius3w/if1004-DevOps

Licença do material

Este Trabalho foi licenciado com uma Licença

Creative Commons - Atribuição-NãoComercial-Compartilhalgual 3.0 Não Adaptada

Mais informações visite

http://creativecommons.org/licenses/by-nc-sa/3.0/deed.pt

Crosscuting Concerns

Security and Security Audits

To err is human; to really screw up you need the root password.
—Anonymous

Introduction

- An initial reaction to discussing security in a DevOps context is to assume that security practices are not agile and can actually hinder improving the time between a code commit and acceptance into normal production
- Discussing adoption of DevOps practices without considering security makes the security team a critic of these practices and dooms the adoption of these practices in many enterprises

Introduction

- Other DevOps activities that are candidates for the discussion of security are
  - Security audits. When a security audit is imminent, coordination between Dev and Ops becomes quite important.
  - Securing the deployment pipeline. The deployment pipeline itself is an attractive target for malicious attackers.
  - Microservice architectures. The adoption of a microservice architecture introduces new security challenges.

What Is Security?

CIA

- Security is easily remembered by the acronym CIA, which stands for confidentiality, integrity, and availability
  - Confidentiality means that no unauthorized people are able to access information
  - Integrity means that no unauthorized people are able to modify information; and
  - Availability means that authorized people are able to access information

Authorization

- The most secure system has no users but also no utility
- It is difficult to allow access by authorized people and deny access by unauthorized people
  - Who is trying to access or modify information and do they have the right to perform the operation they requested?

Defense in depth

- An attacker must circumvent numerous different defenses to compromise your system
- How much your organization is willing to spend on security will depend on how big the loss can be if your system is compromised
  - Equivalently, how much an attacker is willing to spend on compromising your system will depend on the benefit achieved from the compromise

Defense in depth

- Defense in depth also raises the idea that your system can be compromised
- In this case, you will need mechanisms to detect what has happened and to give you the ability to recover
  - Thus, associated with CIA is also a property of nonrepudiation: Individuals cannot deny the operations they performed on the data in your system

Life cycle of an attack

- Attacks can be prevented, detected while they are occurring, or detected after they have succeeded
- Measures to minimize security risks are called security controls
  - Controls can be preventive, detective, or corrective depending on their use within the life cycle of an attack
  - Can be additionally categorized by who is responsible for their implementation

Threats

Threat model

- The point of view of an attacker provides one perspective for you to take when designing your system or subsystem
- Microsoft has introduced the acronym STRIDE for a threat model

Spoofing identity

Tampering with data

Repudiation

Information disclosure

Denial of service

Elevation of privilege

Spoofing identity

- An example of identity spoofing is illegally accessing and then using another user's authentication information, such as username and password

Tampering with data

- Data tampering involves the malicious modification of data

Repudiation

- Repudiation threats are associated with users who deny performing an action without other parties having a way to prove otherwise

Information disclosure

- Information disclosure threats involve the exposure of information to individuals who are not supposed to have access to it

Denial of service

- Denial of service (DoS) attacks target the service availability to valid users—for example, by making a web server temporarily unavailable or unusable

Elevation of privilege

- In this type of threat, an unprivileged user gains privileged access and thereby has sufficient access to compromise or destroy the entire system

Threat model vs CIA

- Spoofing circumvents authentication
- Tampering violates the integrity of data
- Repudiation is an explicit statement of what should happen in the event of a breach or attempted breaking of the rules
- Information disclosure is the negative of confidentiality
- Denial of service compromises availability
- Elevation of privilege is a technique to allow compromise of any of the CIA properties

Insiders

- The Software Engineering Institute (SEI) defines an insider as “a current or former employee, contractor, or business partner who has or had authorized access to an organization’s network, system or data”
- An attack by an insider means that the insider has misused that access to intentionally violate one of the CIA properties

Motives behind an attack

- Financially motivated attacks involve the theft of money or of items that can be sold
- Markets exist for items such as credit card numbers, and tracking those markets can enable you to gain some understanding of the extent of the breaches that have occurred

Motives behind an attack

- Many attacks attempt to gain intellectual property such as trade secrets from commercial organizations or classified information from government organizations

Motives behind an attack

- Sabotage. This category of attacks includes denial-of-service attacks and modifying customer-facing information such as websites as well as out-and-out destruction of sensitive data by disgruntled employees

Resources to Be Protected

- Of the elements of security, C and I refer to “information”
- Information is one of the key resources to be protected
- Information can be at rest, in use, or in transit
- This includes information related to DevOps activities such as source code, test data, logs, updates, and who placed a version into production

Resources to Be Protected

- Information at rest is stored on persistent storage. It can be accessed either through the software systems under the control of one of the actors or through physical possession of the persistent storage
- In the DevOps context, in addition to protecting persistent application-related data, you should consider whether the information you put inside plain-text log lines is sensitive, whether your test data (which could be an early snapshot of sensitive production databases) is well protected, and whether the security around your source code is enough

Resources to Be Protected

- Information in use is being used by an information system. It may be displayed to the user, it may be stored in a cache for performance or reliability reasons, or it may be stored in a virtual machine
- In the DevOps context, many advocate a much shorter server life span (as a result of using phoenix or ephemeral servers, where any change means replacing the old set of VMs through a set of new VMs) for reliability reasons and to prevent server drift

Resources to Be Protected

- Information in transit is being moved from one location to another. If the movement is over a network, then the data is available through access to the network

Computational resources to be protected

- This is the A in CIA
- Authorized users should be able to access the resources they need
- Again, this includes DevOps resources such as the build server
- Resources can become unavailable to an authorized user for multiple reasons

Security Roles and Activities

Security Roles and Activities

- A security architect is responsible for the design of an organization's network to achieve security for the network. The security architect is also responsible for overseeing the implementation of the network
- A solution architect is responsible for the design of systems to support the organization's business functions. Developers implement these designs

Security Roles and Activities

- The organization's IT staff is responsible for monitoring and tracing any events related to potential security attacks. The IT staff is also responsible for the implementation of the architecture designed by the security architect
- The platform provider is responsible for securing the perimeter of the computing platforms used by an organization, ensuring isolation among the customers of the platform, and ensuring that the customers get adequate resources when they require them. The platform provider also provides services used by the security architect.

Identity Management

Identity Management

- Identity management refers to all tasks required to create, manage, and delete user identities
- All activities within the identity management task should be logged for audit purposes—not only human-initiated activities but also activities performed by tools or scripts
- Invocations among (micro)services also need to be authenticated

Identity Management

- The platform provider provides the means to manage the identity of all of the users of the platform, and the security architect does the same for all users of an organization’s systems

Authentication

- Authentication controls are intended to verify that you are who you say you are. That is, they protect against a spoofing attack
- Authenticating an individual gets complicated for a number of reasons
  - “You” may not mean you, but may mean a system operating on your behalf
  - “You” may not be uniquely identified by the system, but instead “you” may be a role
  - Your authentication mechanism (e.g., password or certificate) may have been compromised
  - You may no longer be an employee or authorized user of the system

Controls Relating to a System Operating on Your Behalf

- Hardware
- Software

Role-Based Authentication

- Is a technique for assigning identifications based on roles rather than on identity
  - The problem with RBA is that there is no traceability

Controls to Prevent Compromising Passwords

- An attacker breaks an individual's password through various forms of brute force attacks
- A user allows her or his password to be determined through social engineering means
- An authorized user changes roles or leaves the organization
- Your system is compromised, allowing determination of passwords

Authorization

- Once a user is identified then it becomes possible to control access to resources based on the privileges granted to that user

Techniques to Control Access to Resources

- Access Control Lists. An ACL is a list of users or roles and allowed operations attached to a resource such as a file system or database field
- Capability. A capability is a token that grants particular rights on a resource. A good analogy is a key and a lock

Role-Based Access Control

- RBAC is based on a mapping between individuals and roles
- A role is allowed certain access privileges, and the identity management system maintains a mapping between users and roles
- It also maintains a mapping between roles and privileges
- Then, when a user changes roles, the mapping between users and roles is changed as well and the authorization system is provided with the information appropriate to the new role

Access Control

Access Control

- Preventing Access
- Defining Boundaries
- Isolation
- Encryption
- Decommissioning Data
- Patching
- Change Management

Detection, Auditing, and Denial of Service

Detection, Auditing, and Denial of Service

- The R in STRIDE stands for repudiation. Both for business reasons (e.g., “I did not order that”) and forensic reasons (e.g., “What damage did the attacker do?”) auditing activities are important
- Once an audit trail has been created, it must be protected
  - must be encrypted, stored independently of the systems that are being audited, and have protected access
- Audit trails != logs

Detection, Auditing, and Denial of Service

- The one element of STRIDE we have yet to discuss is the D—denial of service
- Denial-of-service protection is the responsibility of the platform provider and the security architect
- A variety of technologies and tools exist to limit the effect of denial-of-service attacks

Development

Development

- Controls exist in NIST 800-53 that specify aspects of the development process
- Infrastructure-as-code, scripts and other inputs into DevOps tools must be developed and should be subject to the same scrutiny as application code development
- On the other hand, security testing must be integrated with the deployment pipeline

Five design principles for security

I. Provide clients with the least privilege necessary to complete their task. If temporary access is needed it should be rescinded right after use

II. Mechanisms should be as small and simple as possible

III. Every access to every object must be checked not only during normal use but also during initialization, shutdown, and restart

IV. Minimize the number of mechanisms common to more than one user and depended on by all users. Every shared mechanism is a potential information path

V. Utilize fail-safe defaults. Argue why a particular process or client needs to have access, not why that process or client should not have access

Five design principles for security

- These design principles apply to both the application design and the deployment pipeline itself
- Security is more than a matter of a good design; it is also a matter of good coding practices

Auditors

Auditors

- With this background, what does an auditor look for?
- The answer is "all of the above"
- An auditor should want to consider everything from development practices on code and scripts to which controls are used to protect against what kinds of attacks

Application Design Considerations

Application Design Considerations

- The use of the cloud and microservice architecture leads to some special design considerations for security
  - A few additional security considerations must be taken for the application host, i.e. the VMs in the cloud
  - Components should be able to be isolated and deployed independently without affecting other components
  - Components should be coded to be defensive and not to trust their invoker
  - Components are provided with configurations appropriate to the environment in which they are executing
  - Configurations should be saved in version-controlled persistent storage
  - Invocation among services should be authenticated, with performance penalties of authentication being one of the considerations
  - Communication to the external world should be encrypted, and communications among internal services should consider encryption
  - Use a well-patched base image to create other customized images for individual microservices so the attack profiles can be reduced

Deployment Pipeline Design Considerations

Deployment Pipeline Design Considerations

- The deployment pipeline itself can also be hosted in the cloud
- Especially the testing environment, which can benefit from cloud elasticity, repetitive clean setup, and better consistency among different environments
- The cloud hosting security considerations will be the same as we mentioned in the previous section

Security Considerations

- Lock down your pipeline environment most of the time and track all changes to the pipeline
- Integrate continuous security testing throughout the pipeline, which includes IDE/pre-commit analysis, build and integration servers, and end-to-end testing environment
- Integrate security monitoring in the production environment
- Tear down testing environments every time the respective tests are finished, or at least regularly
- Consider encrypting sensitive logs and test data, both at rest and in transit

Security Considerations

- Automate the pipeline as much as possible through infrastructure-as-code, and promote code reuse, especially for improving environment consistency between various testing and production environments
- No direct change is permitted to any of the environments without going through the pipeline (and its change tracking)
- Test your infrastructure code (not just application code) for security vulnerabilities
- Be able to generate regular conformance and auditing output though automation

For Further Reading

- For cloud-specific security issues, you can find an extensive catalogue of patterns at http://bit.ly/2DCbeYz
- You can find more about the STRIDE threat model at http://bit.ly/2xFZYF8
- For ways of mitigating insider attacks, the Software Engineering Institute has a technical report at http://bit.ly/2NKI8Ms
- NIST 800-53 is a catalogue of security controls for U.S. federal information systems. You can find it and many related publications at http://bit.ly/2DAauDc
- A discussion of the BrowserStack attack can be found at http://bit.ly/2DzcOdD
- Security Monkey is described at http://bit.ly/2zzDWp0