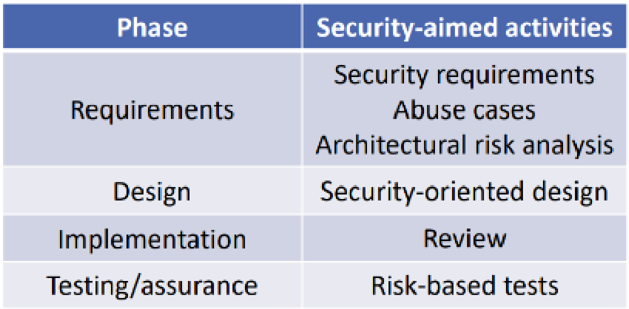
SECURITY-ORIENTED DESIGN AND ARCHITECTURES

DESIGNING SECURITY AWARE SYSTEMS

There are genrally 2 different approachs to design security systems

- flawed approach where we design, develop and deploy the system but we ignore the security at first
- better approach where we focus on security from the start



Phase	Security-aimed activities
Requirements	Security requirements Abuse cases Architectural risk analysis
Design	Security-oriented design
Implementation	Review
Testing/assurance	Risk-based tests

SECURE HARDWARE

In some cases is possibile to introduce the security directly in the harware, for example we can have:

- Intel AES-IN implements cryptography instructions
- Intel Software Guard Extensions (SGX) support “encrypted computation” for cloud computing application

Another technique is to use harware primitives for security, where we can find:

- Physical Unclonable Functions,
- Intel Memory Protection Extensions (MPX)

THREAT MODEL DRIVEN DESIGN

In general different threat models will elicit different responses and bad model implies bad security.

Any assumption we make in our model are potential holes that the adversary can exploit.

Example of bad assumptions:

- Encrypted traffic carries no information and this is not true, by analyzing the suzr and distribution of messages we can infer application state

FINDING GOOD THREAT MODELS

The basic idea is to compare our system against similar ones and look at what their design can mitigate.

Undestand and study past attacks and attack patterns.

Challenge assumptions in your design

DEFINING SECURITY REQUIREMENTS (POLICIES)

Here there are some different approaches to perform this task:

- Approach 1:
  - follow the standards and regulations
- Approach 2:
  - policy arising from threat modeling

DEFINING ABUSE CASES

As opposed to use cases that is what a system should do the abuse cases are used to describe what a system should not do.

Example of abuse cases:

- A user is able to spoof being a manager and thereby change the interest rate on a bank account

Using attack patterns or scenarios we can construct abuse cases in which an adversary can violate a security requirement.

DESIGN PRINCIPLES

A principle is a high-level design goal with many possible manifestations

Prevention principles

The goal is to eliminate defects entirely

Mitigation principles

The goal is to reduce the harm from exploitation of unknown defects

Detection (and recovery) principles

the goal is to identify and understand an attack and undo damages

SOME DESIGN PRINCIPLES

FAVOR SIMPLICITY

Keep it simple, do not expect expert users.

Use fail-safe default. This because some configuration or usage choices affect a system's security (for example the length of the cryptographic keys, the choice of a password etc).

In general the default choices should be secure (for example for RSA the default key is 2048-bit), and also the using of whitelist valid objects rather than a black list is a good choice.

TRUST WITH RELUCTANCE

Employ a small trusted computing base.

For example the operating system kernels must be minimized in order to reduce the trusted components. In some cases the device drivers are moved outside of the kernel when we use a micro-kernel design (safe thing)

Grant the least privilege possible

Avoid to give a part of system more privileges than it needs to do its job.

Input validation is a kind of least privilege because we trust a subsystem only under certain circumstances.

Sandboxing is also a kind of trust with reluctance, we isolate a system component reducing its privilege by making certain interactions impossible.

DEFEND IN DEPTH

Apply security by diversity, so if a layer is broken there is another that needs to be bypassed.

Use community resources and avoid to implement critical functionalities by scratch. Is better to reuse well-known libraries.

MINITOR AND TRACE

Software must be designed to log relevant operational information.

BUSTING SECURITY MYTHS

“The vulnerability is not in the OWASP top ten”

Just because a vulnerability is not in the top 10 list does not mean you should ignore it, for example:

- **buffer overflow,**
- **null pointer exceptions etc**

are not in the top 10.

“The site is secure because we use HTTPS”

The fact is that application flaws can still be exploited, in fact also if the data is malicious it will transit safely with HTTPS

“robots.txt will disallow the access to files”

The robots.txt file is not a security control, it just tells web crawlers what they have to index and not to index.

“We are secure because we use blacklists”

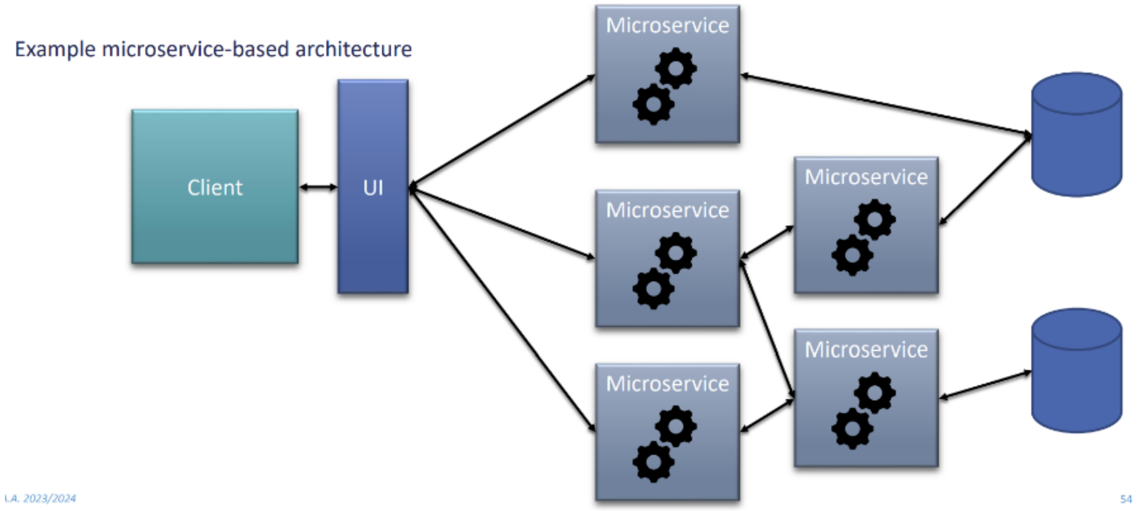
There are some cases in which the blacklist doesn’t takes in count some malicious aspects because they are not added.

So is better to use whitelists rather than blacklists.

“The newest release is more secure”

It is not true, for example it can be that the new release introduces a bug.

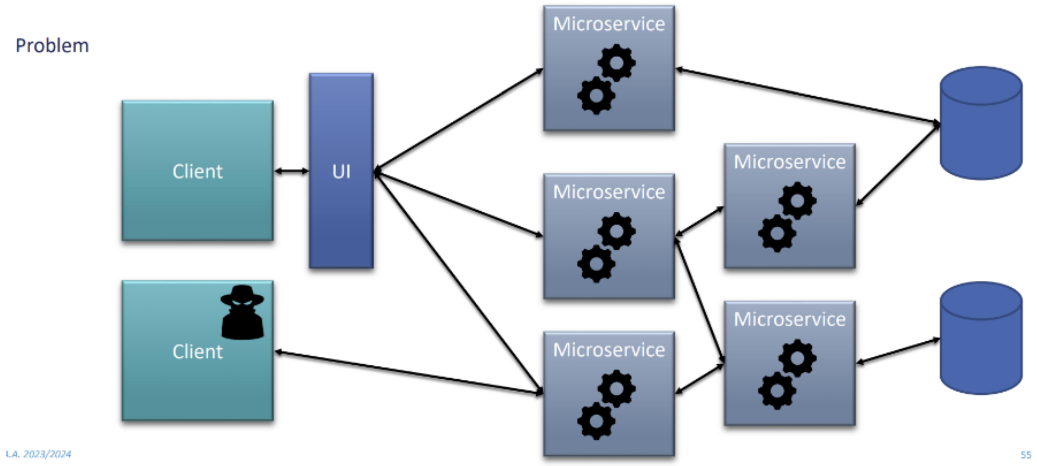
“My API is in the back end so no body can access it.”



Example microservice-based architecture

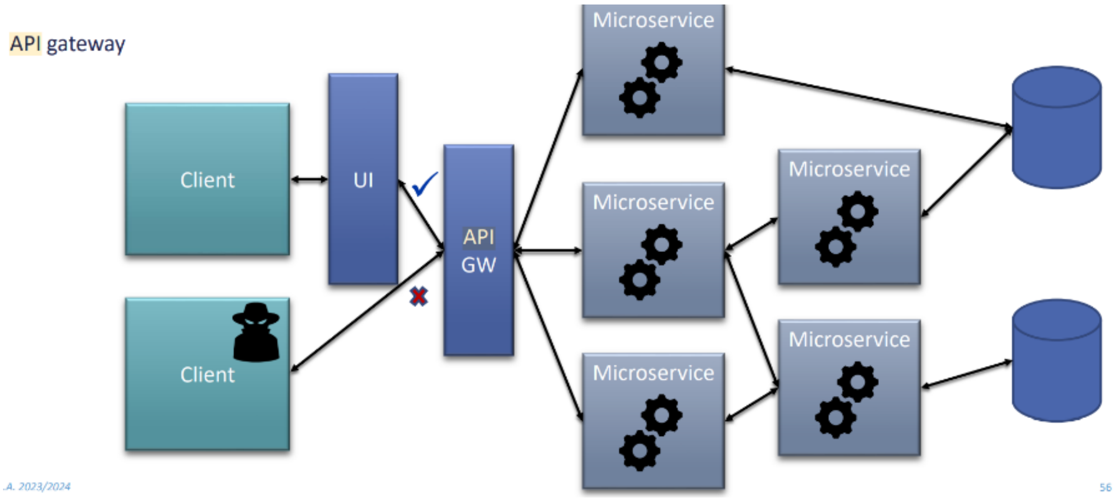
I.A. 2023/2024

54



Problem

One solution here is to use a API gateway that operates as a firewall when we try to access to the API’s endpoints.

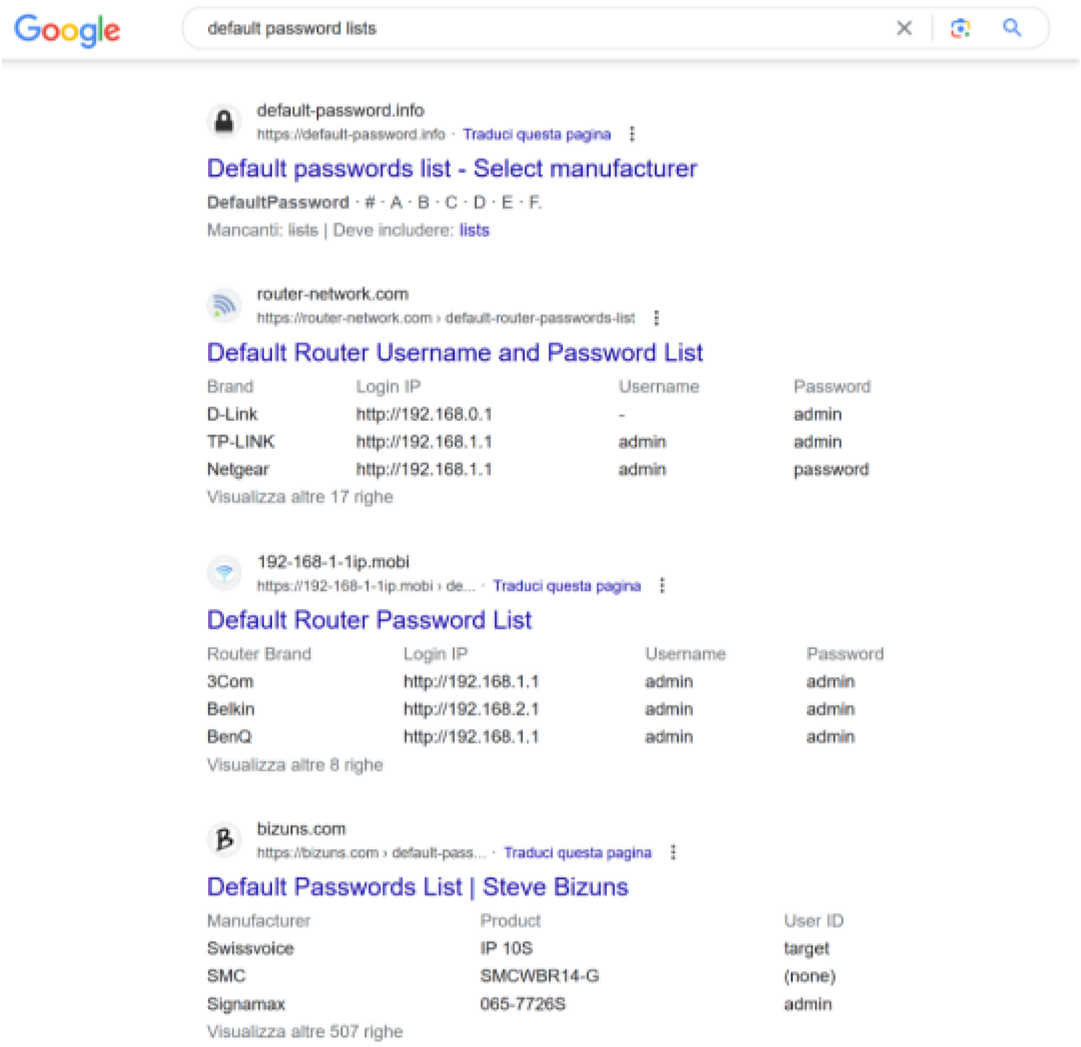


API gateway

“No security configuration needed.”

Unless security validation is applied the plug-and-play approach increases the attack surface.

For example leave the router default password can be a dangerous thing because it can be predictible.



default-password.info
https://default-password.info · Traduci questa pagina

Default passwords list - Select manufacturer

DefaultPassword · # · A · B · C · D · E · F.

Mancanti: lists | Deve includere: lists

router-network.com
https://router-network.com › default-router-passwords-list

Default Router Username and Password List

Brand	Login IP	Username	Password
D-Link	http://192.168.0.1	-	admin
TP-LINK	http://192.168.1.1	admin	admin
Netgear	http://192.168.1.1	admin	password

Visualizza altre 17 righe

192-168-1-1ip.mobi
https://192-168-1-1ip.mobi › de... · Traduci questa pagina

Default Router Password List

Router Brand	Login IP	Username	Password
3Com	http://192.168.1.1	admin	admin
Belkin	http://192.168.2.1	admin	admin
BenQ	http://192.168.1.1	admin	admin

Visualizza altre 8 righe

bizuns.com
https://bizuns.com › default-pass... · Traduci questa pagina

Default Passwords List | Steve Bizuns

Manufacturer	Product	User ID
Swissvoice	IP 10S	target
SMC	SMCWBR14-G	(none)
Signamax	065-7726S	admin

Visualizza altre 507 righe

SECURITY ARCHITECTURES

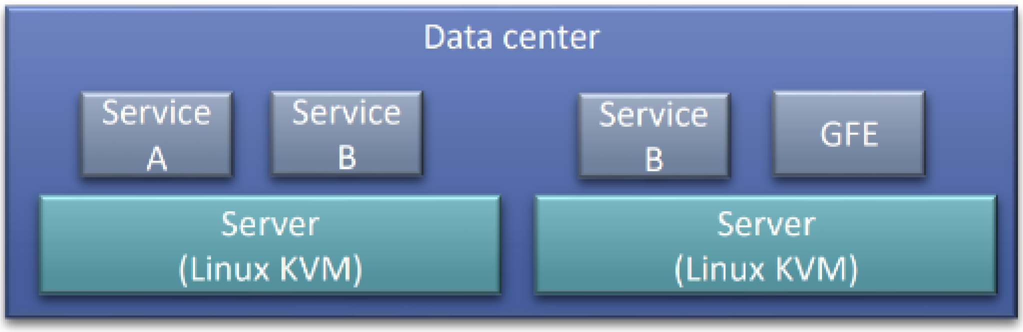
Google security architecture

The google security architecture is based on a platform where we have:

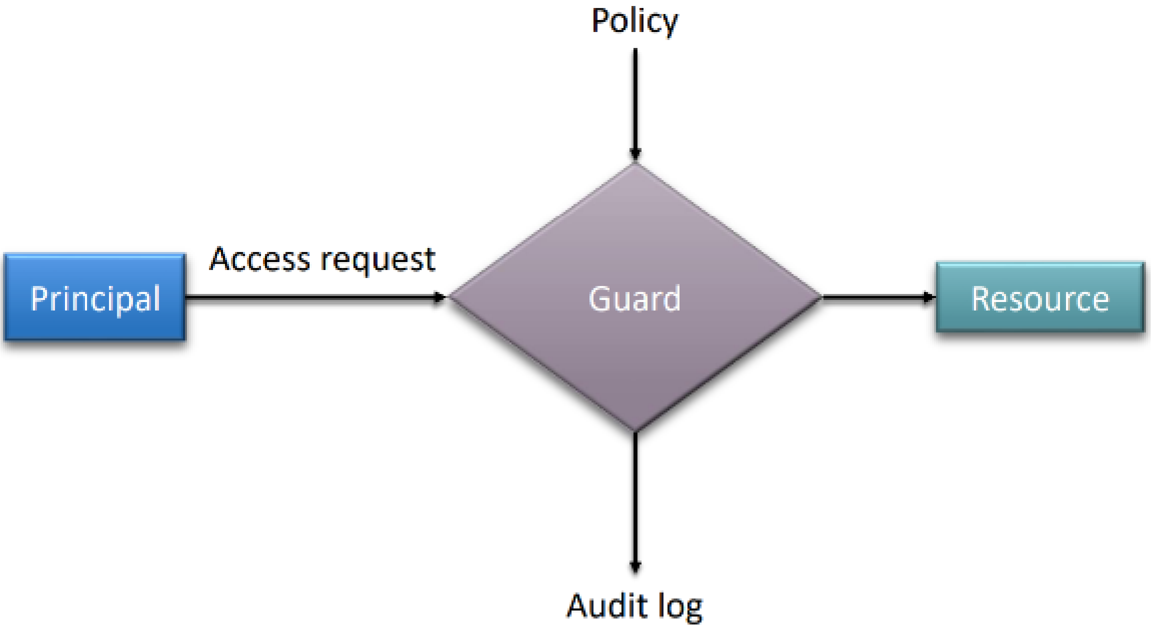
- **a pool of interconnected data centers**

All applications and high-level services are mapped to lower level services.

Google Front-Ends (GFE) handle incoming Https connections and turns them into a more structured and secure format.



The data sharing is based on the using of a reference monitor called Guard:



The principals of this system are the end users, google employees, services and the machines.

The resources that can be shared are data, network, CPU and memory.
The main activity performed by the guard is:

- **authenticate the principal**
  - it is done for the humans with 2 FA, for services with a public key
- **authorize the resource access**
  - it is generally based on the using of ACLs where for each resource we have a list of (principal, permissions)
- **log the operations**
  - for every action we store the timestamp, the principal, the resource accessed and the action performed.

HARDWARE DEFENSE

For the hardware defense google designed their own motherboards and security chips that are used on the boot and checks that the BIOS and the OS are signed by Google's private key.