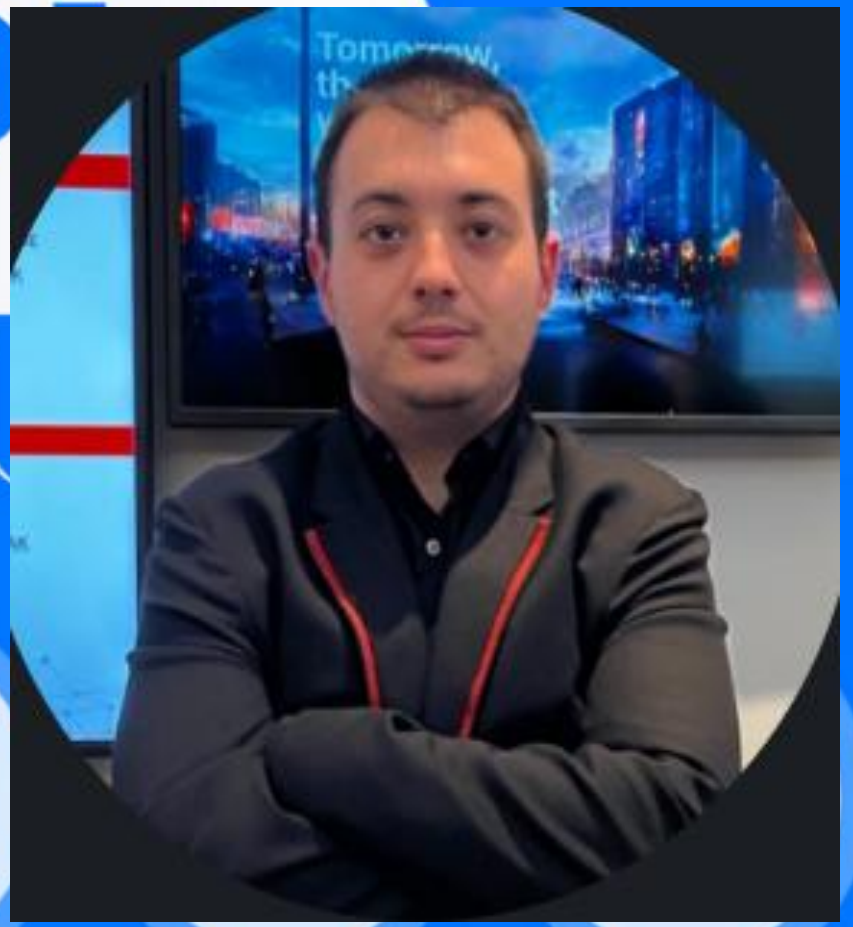
OWASP 2023 TOP 10 API Vulnerabilities and mitigations

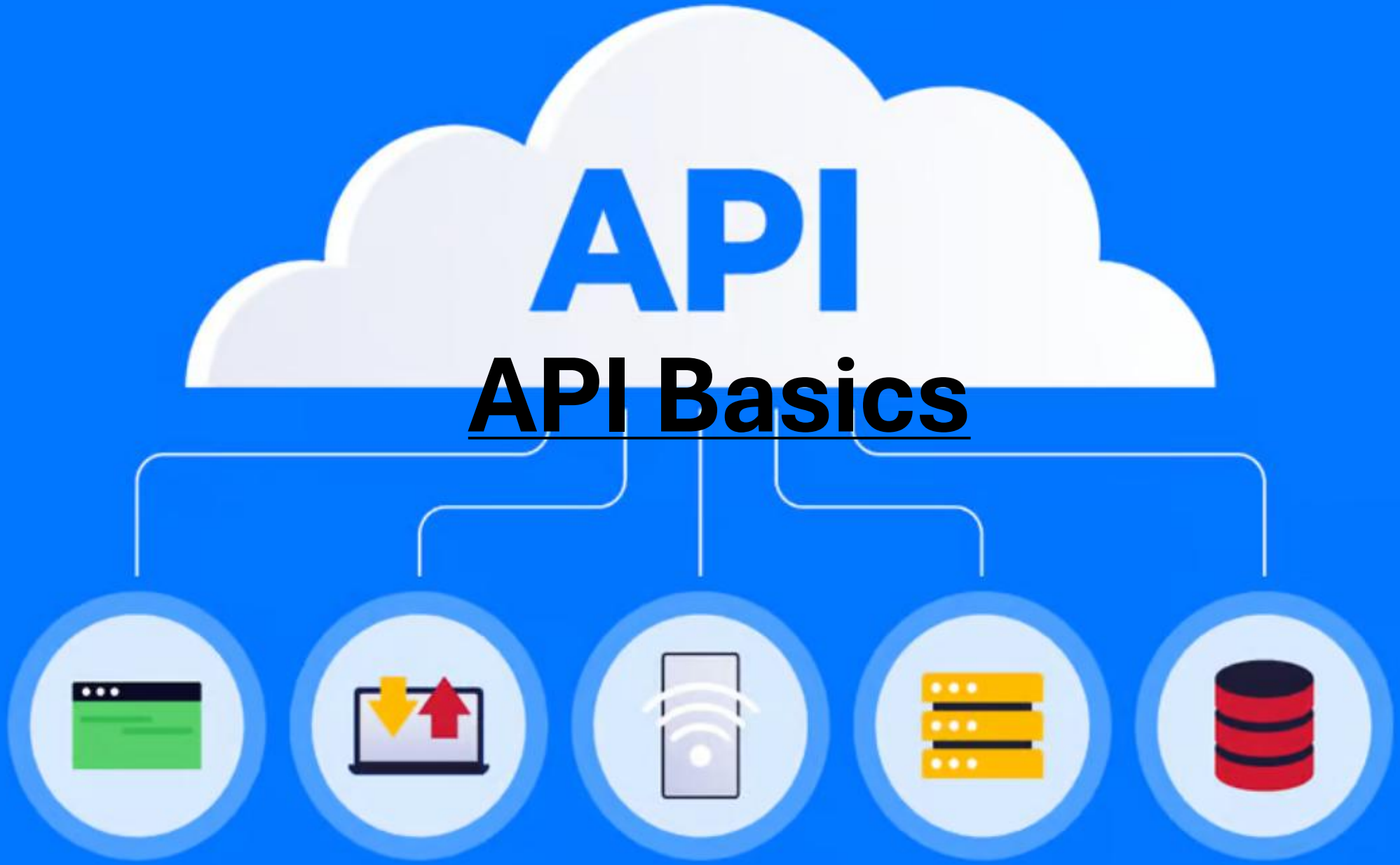
About me:

- Network Security and Cybersecurity consultant with more than 10 years of experience in Networking, Network Security and Cybersecurity.
- Holder of Comptia Pentest+ and Comptia CASP+ certifications and many other IT vendor sertifications.
- Master’s degree in Telecommunications.



Content:

- API Basics
- Top 10 API vulnerabilities and their mitigation
- Demo Session



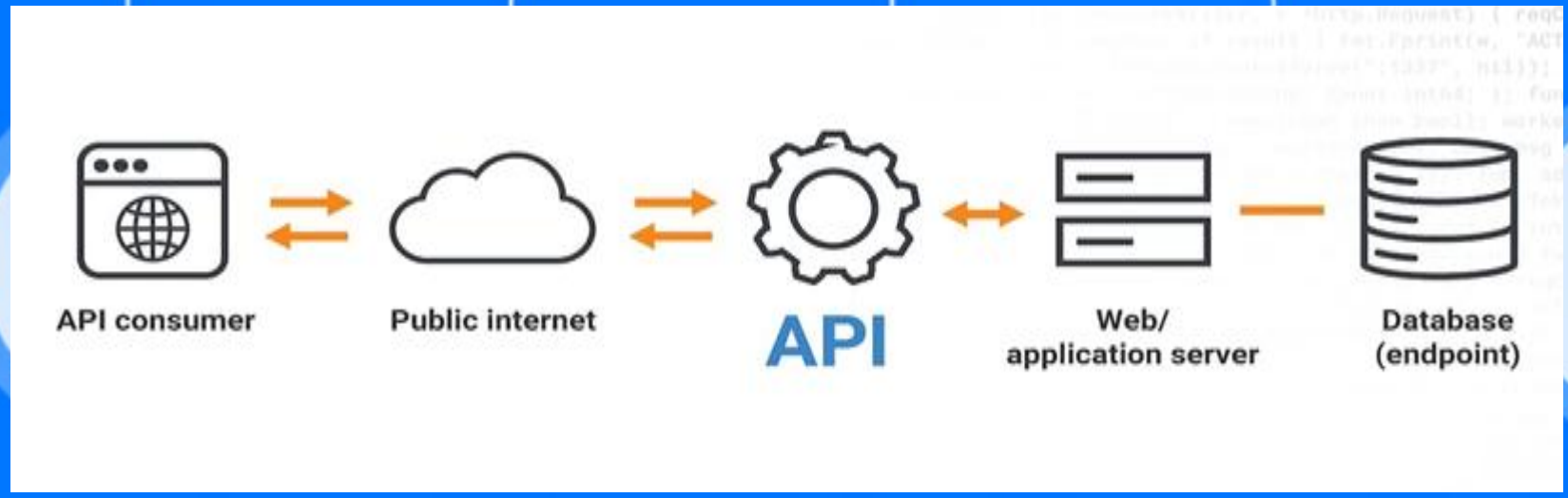
API Basics

API Basics

Application programming interface (API) is a connection between computers or between computer programs.

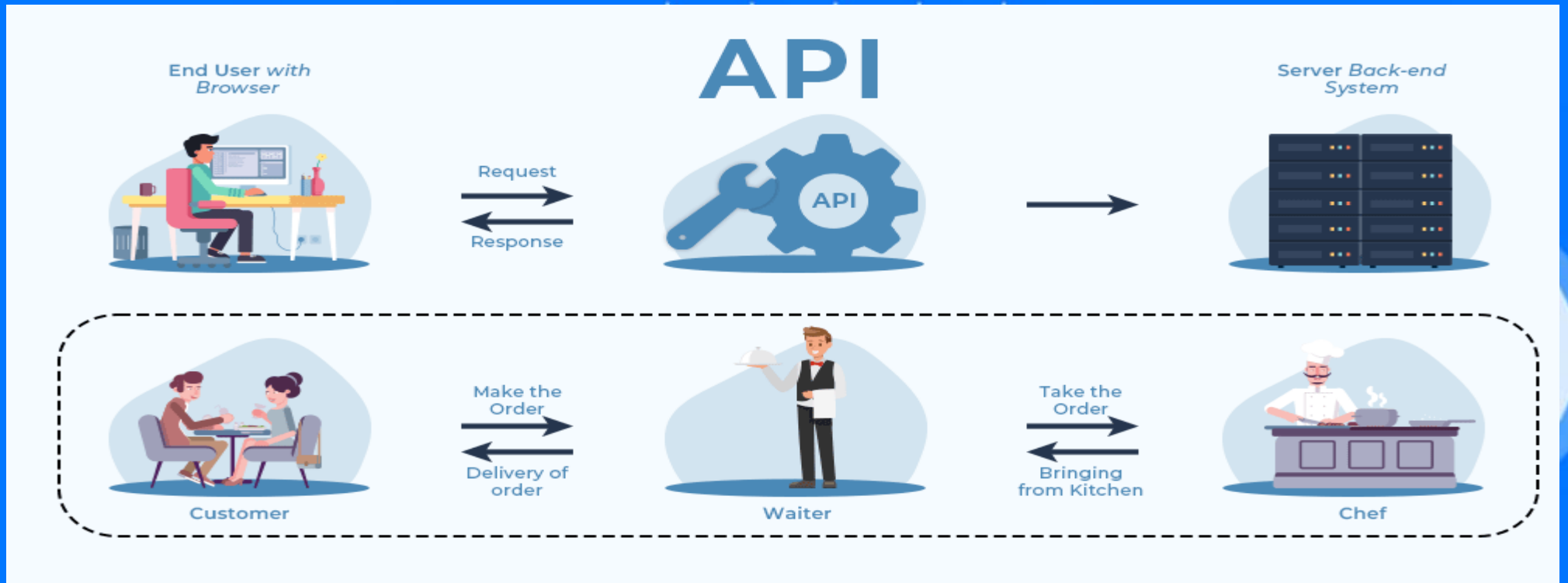
Nowadays API communication is even more common than the communication between users with web browsers and custom app agent clients (usually when using mobile phones) and the web applications.

Many Web applications integrate 3rth party functions, using API, like for example subscribing and ingesting weather report information, using API and so on.



API Basics

API traffic is different than the normal web traffic, even if in most cases it uses the same underlying protocols as HTTPS/HTTP. API clients usually do not support java scripts or cookies, so attacks based on JavaScript injections or cookie manipulations do not affect them, but this is also true that defenses based on javascripts or cookies also can’t be used!



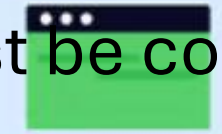
API Basics

API data is usually in XML or JSON formats.

The newer format is JSON, so it is currently more prevalent.

The different API data formats have different vulnerabilities and attacks that can make use of those vulnerabilities, so this must be considered.

There is even newer API format, called GraphQL!



XML

```
<empinfo>
   <employees>
      <employee>
         <name>James Kirk</name>
         <age>40></age>
      </employee>
      <employee>
         <name>Jean-Luc Picard</name>
         <age>45</age>
      </employee>
      <employee>
         <name>Wesley Crusher</name>
         <age>27</age>
      </employee>
   </employees>
</empinfo>
```

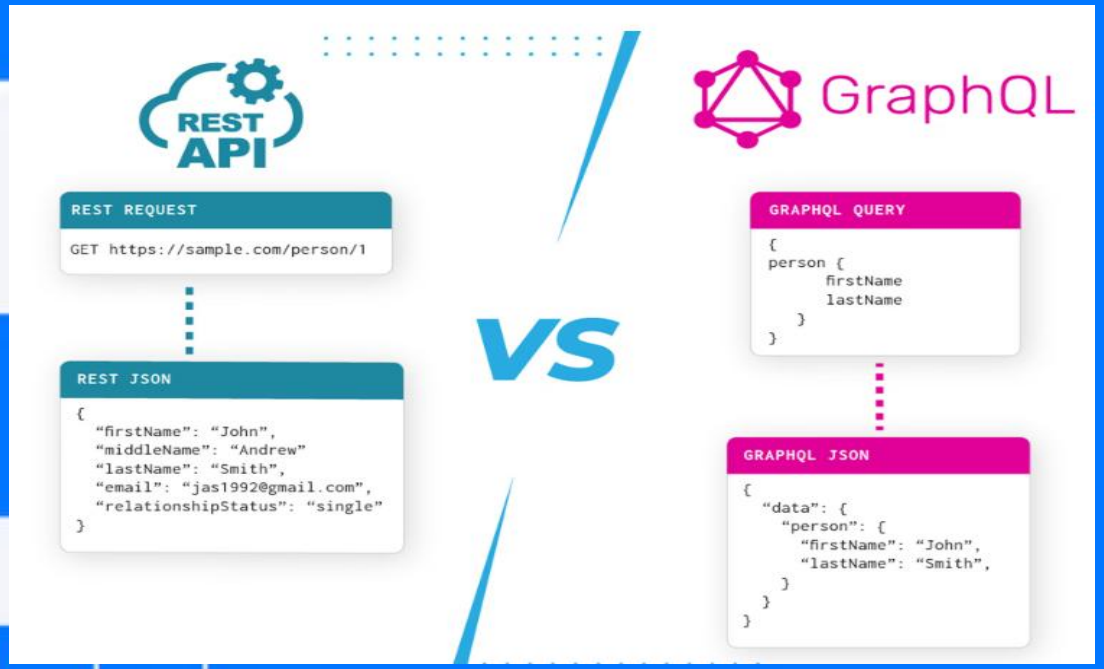
JSON

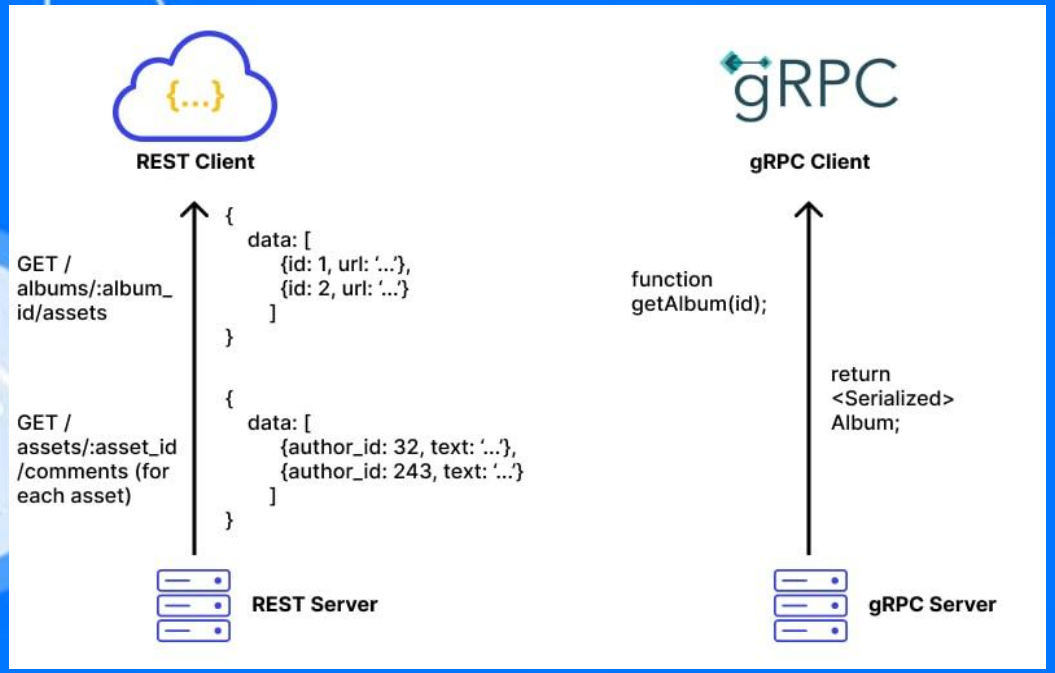
```
{  "empinfo" :
      {
            "employees" :  [
            {
                  "name" : "James Kirk",
                  "age" : 40,
            },
            {
                  "name" : "Jean-Luc Picard",
                  "age" : 45,
            },
            {
                  "name" : "Wesley Crusher",
                  "age" : 27,
            }
                              ]
      }
}
```

API Basics

SOAP API was the old API model, that used XML for data format while the current 2 new API models are GraphQL and REST API and they use JSON. GraphQL less URL endpoints as it uses HTTP POST body that has query section that describes the requested resource and offers it's own security risks and benefits.

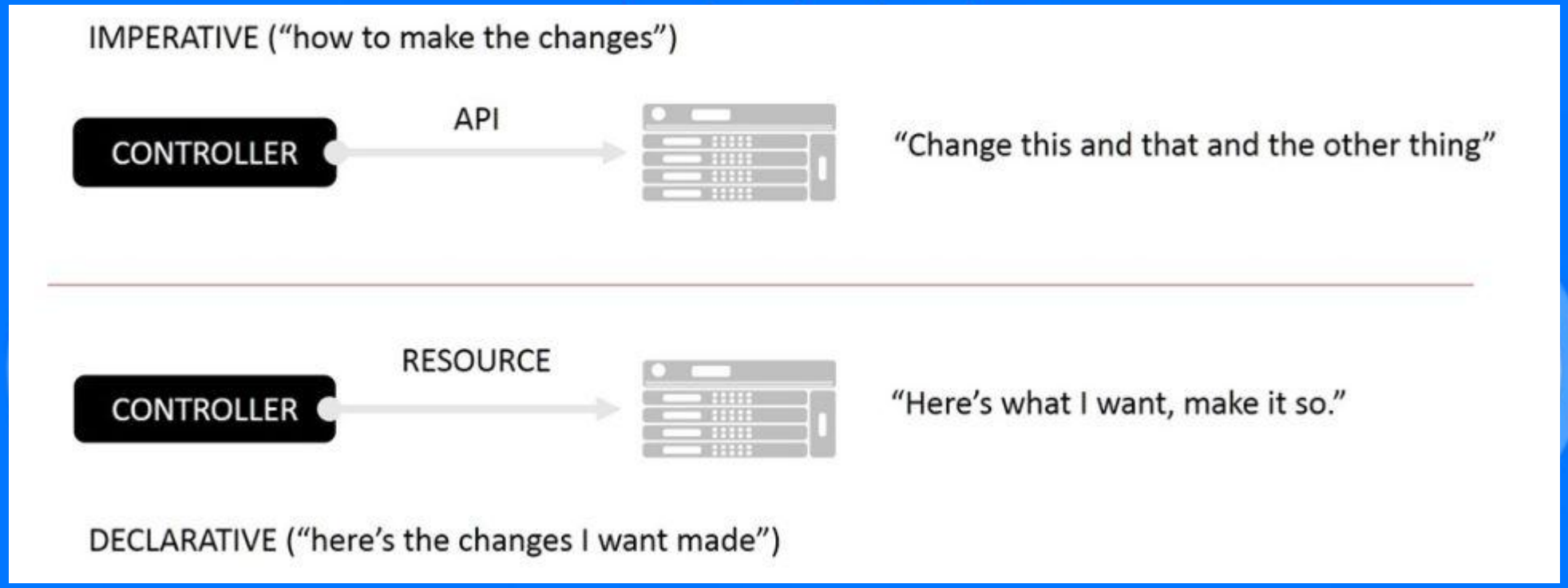
qRPC is a new way to send function calls remotely but it will take time this to become popular and this API is used most internally in a microservice between the components. I see this being a big place for attacks in the future ☺





API Basics

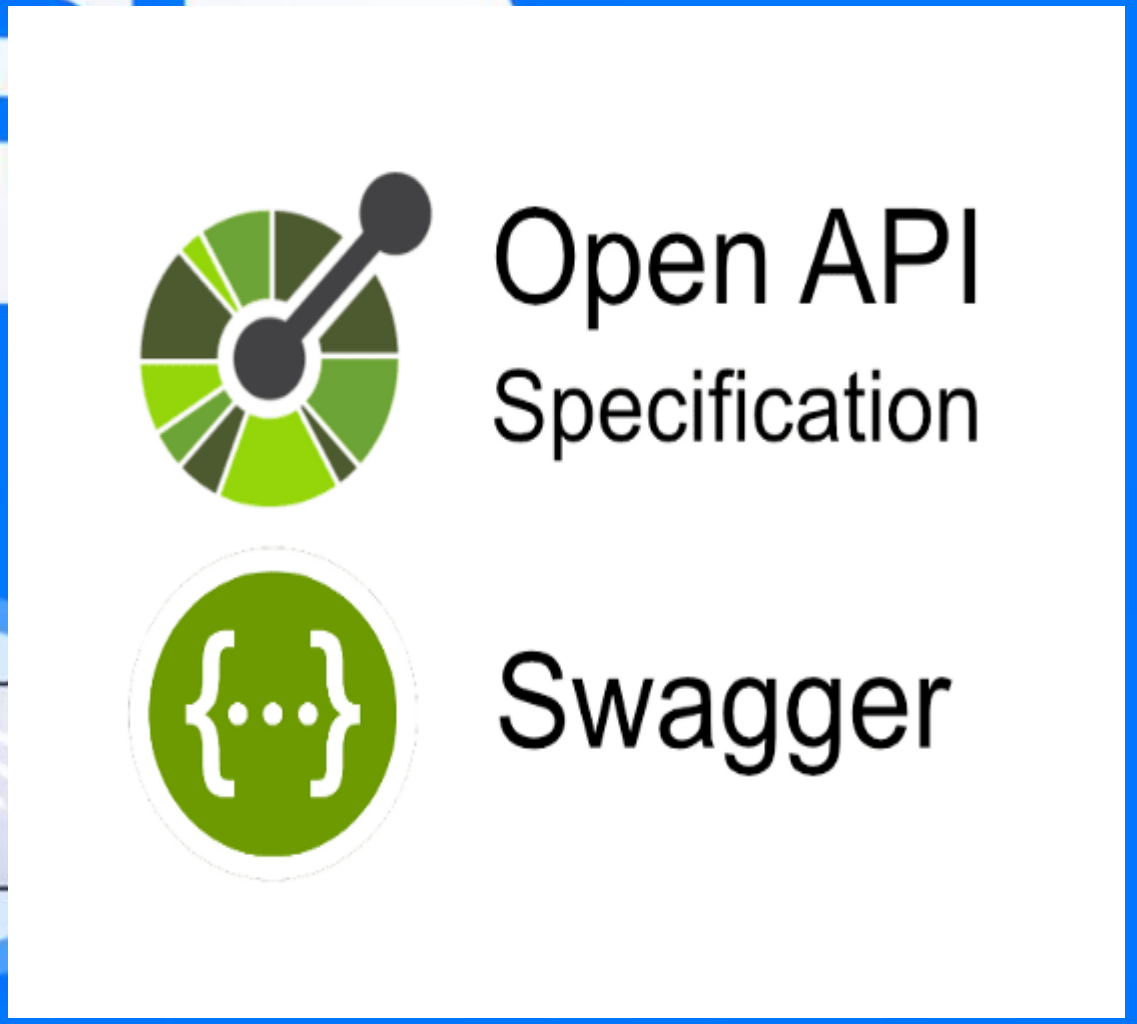
Even not so much related to security nowadays there are two API models called imperative and declarative API. The difference is that for declarative API that is the newer less domain knowledge is needed of the end system to configure it as the declaration message declares what the controller expects and not a specific command.



API Basics

API endpoints that are also the url's we use in our web browsers are well documented in the form of API specification that is called Swagger(OpenAPI2.x) or OpenAPI3.x.

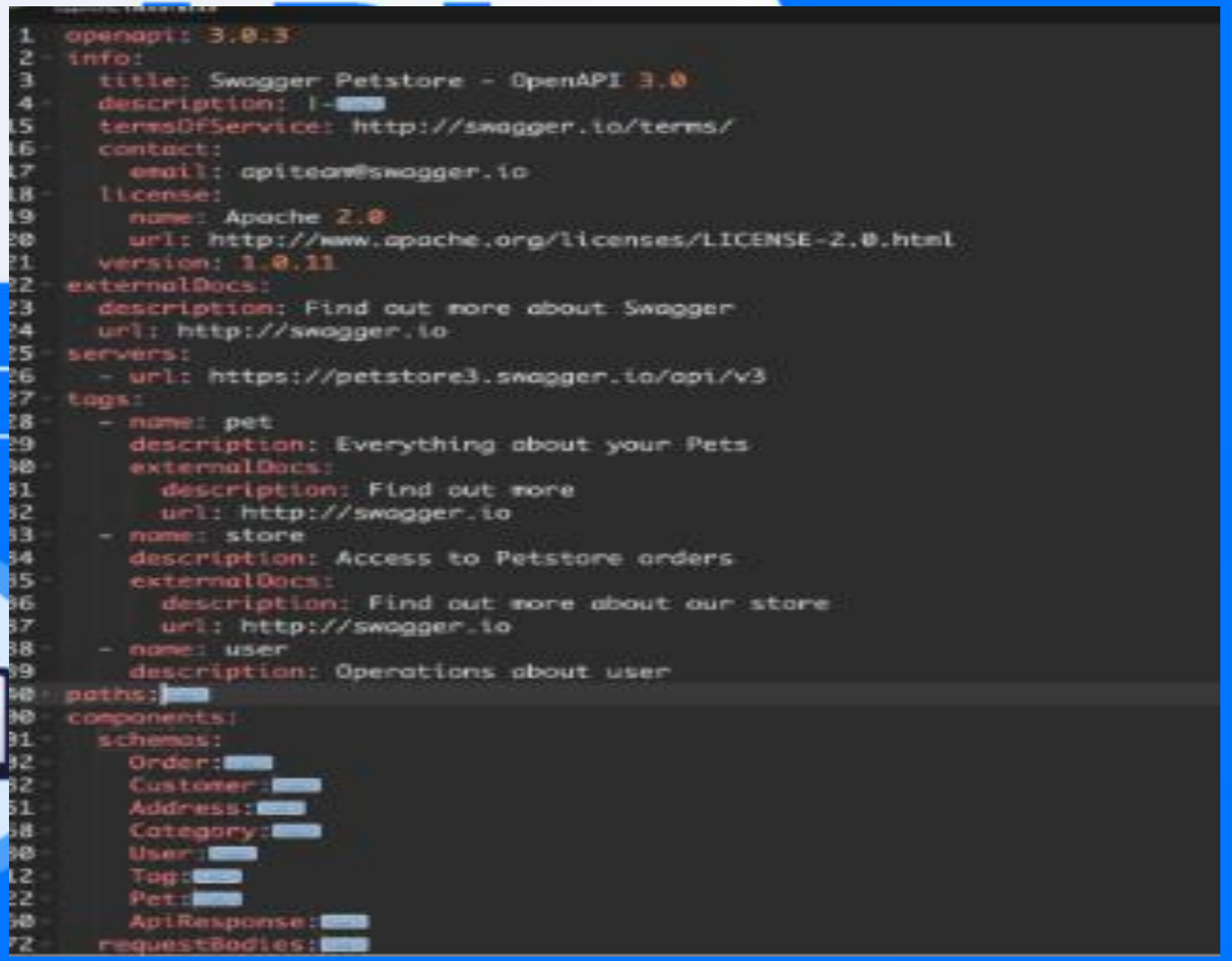
As there are instances where streaming bidirectional API traffic is needed, where the servers can also initialize requests, there is a specification for this kind of communication, called AyncAPI. Protocols that can make use of this are WebSocket or MQTT for IOT devices API communication.



API Basics

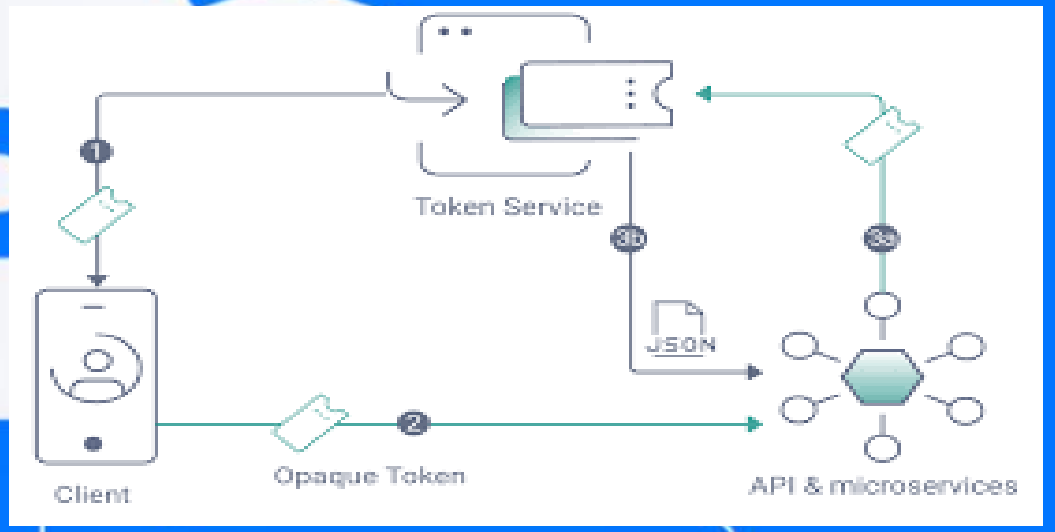
OpenAPI specifications are used, so that developers that can use the API to know how program their API application client, so it is for of API documentation.

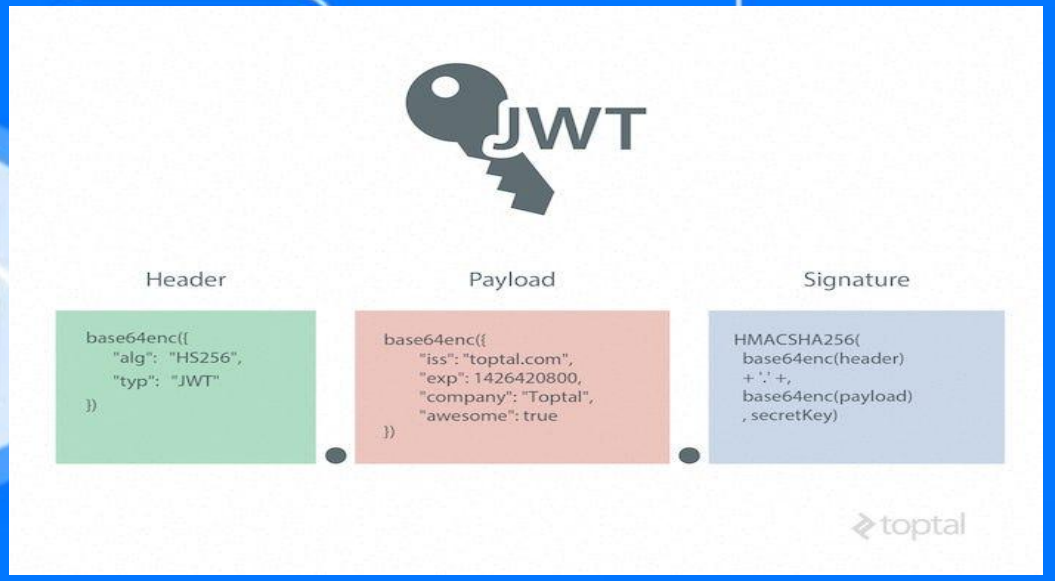
The secondary purpose is for security as the spec can describe which endpoints exist(this protects again so-called shadow API undocumented endpoints), what headers and parameters should exist in the request and what should be their value and even what is Security Authentication for the API!



API Basics

API Authentication and are authorizations in most cases are different than the normal web authentication,. For example, session cookie that is used after the initial user authentication in most cases can't be used. Old ways to authenticate API and users was the basic authentication for every request but nowadays things like opaque and JWT tokens are used! JWT CA certificate signed token is given after the initial authentication, and it holds information that not only allows the API access but also enforces authorization using claims that can be checked if the users should have access to an explicit API endpoint. Opaque tokens are just random strings that have to exchanged with AS (Authorization server) to get the API access levels in the form of another token, called introspect token.

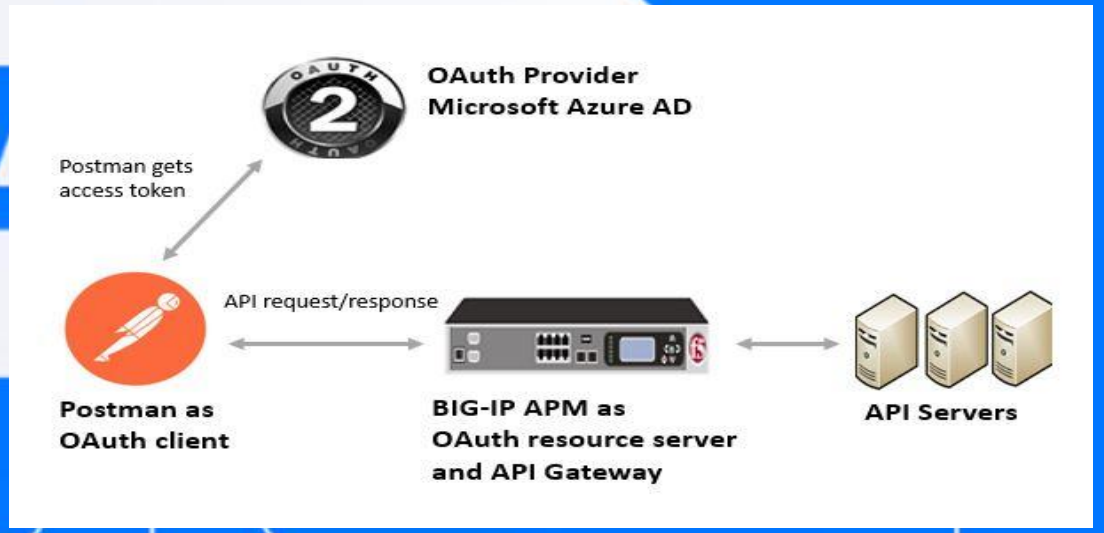


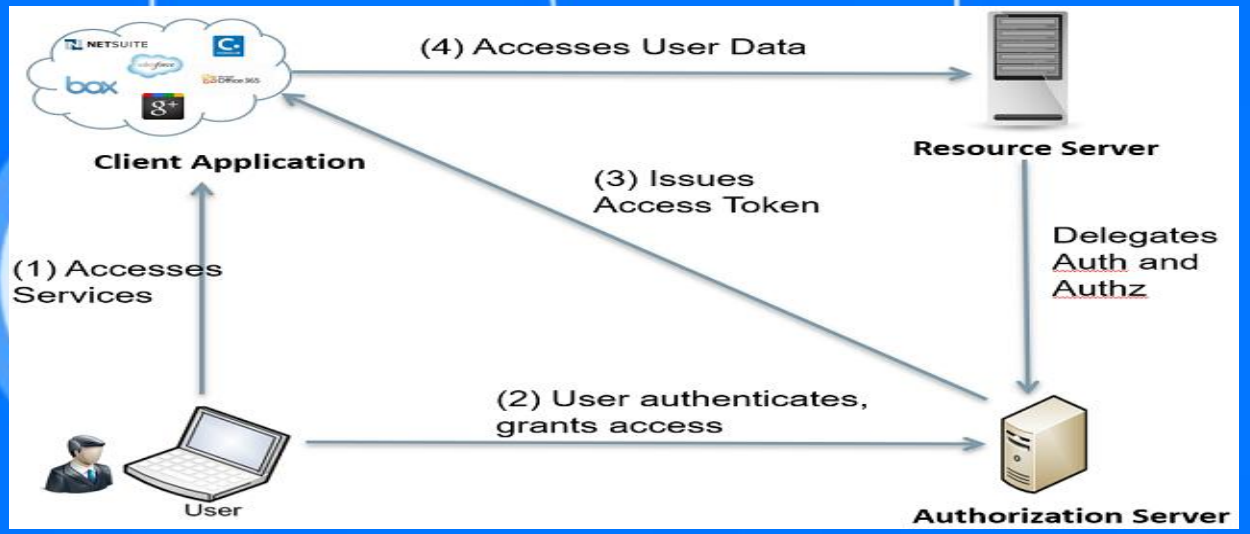


API Basics

OAuth2.0 is one of the most popular protocols for JWT or Opaque tokens.

OpenID is an authentication protocol used for signing users into client applications. The purpose is user authentication. OAuth is an authorization protocol used for providing client applications delegated access to server resources on behalf of a user. The purpose is delegated authorization. OpenID is an extension for users to trigger to apps to use Oauth.

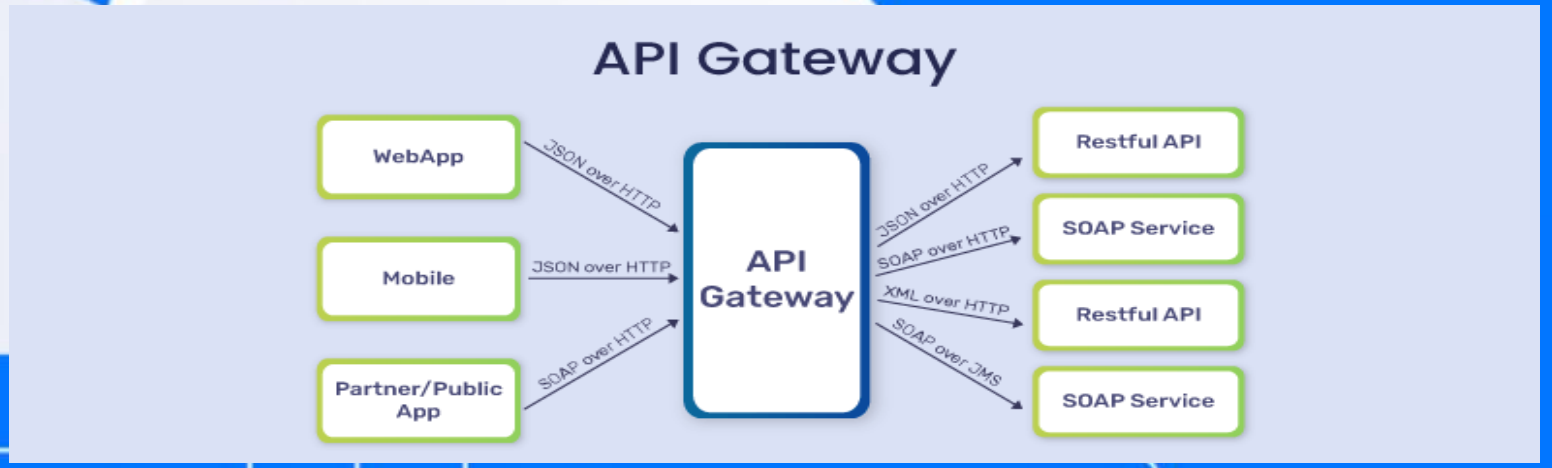


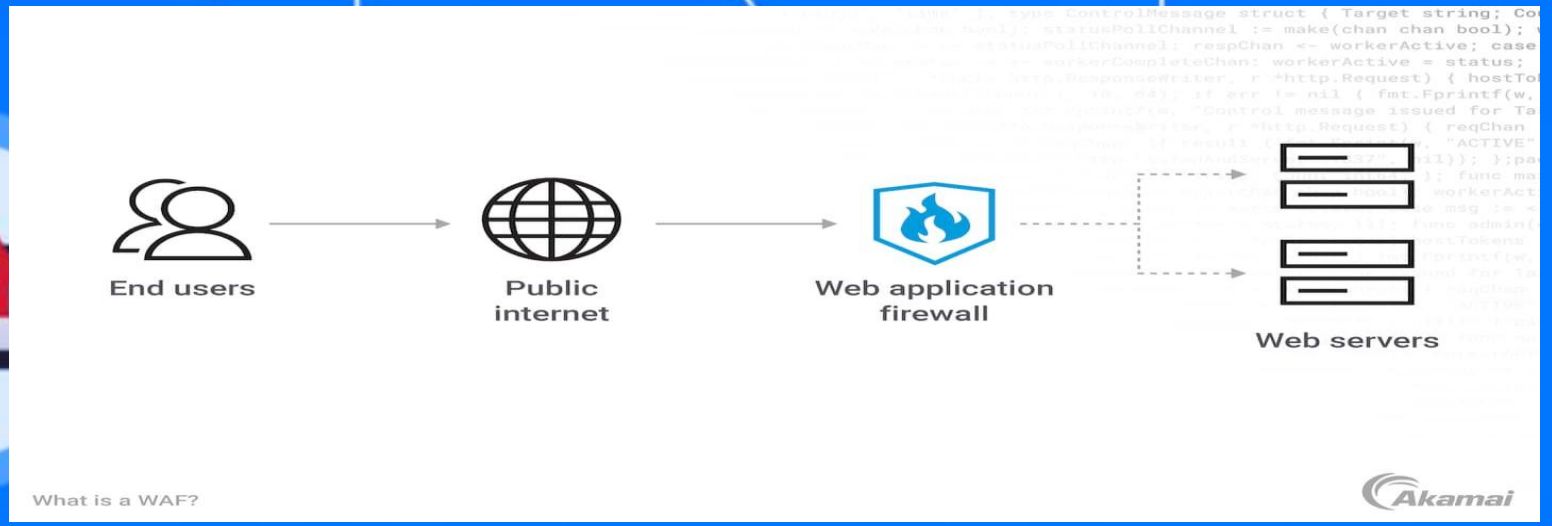


API Basics

Usually for good API security devices added. API Gateways can translate between internal and external API protocols, enforce authentication and authorization of the API traffic with federation services like for example Azure AD and even enforce the OpenAPI scheme. API Gateways can also translate between external protocol like REST API and internal, for example qRPC

Web Application Firewalls can provide signature protection, DOS and rate limiting for the API traffic, like they do for normal user traffic.

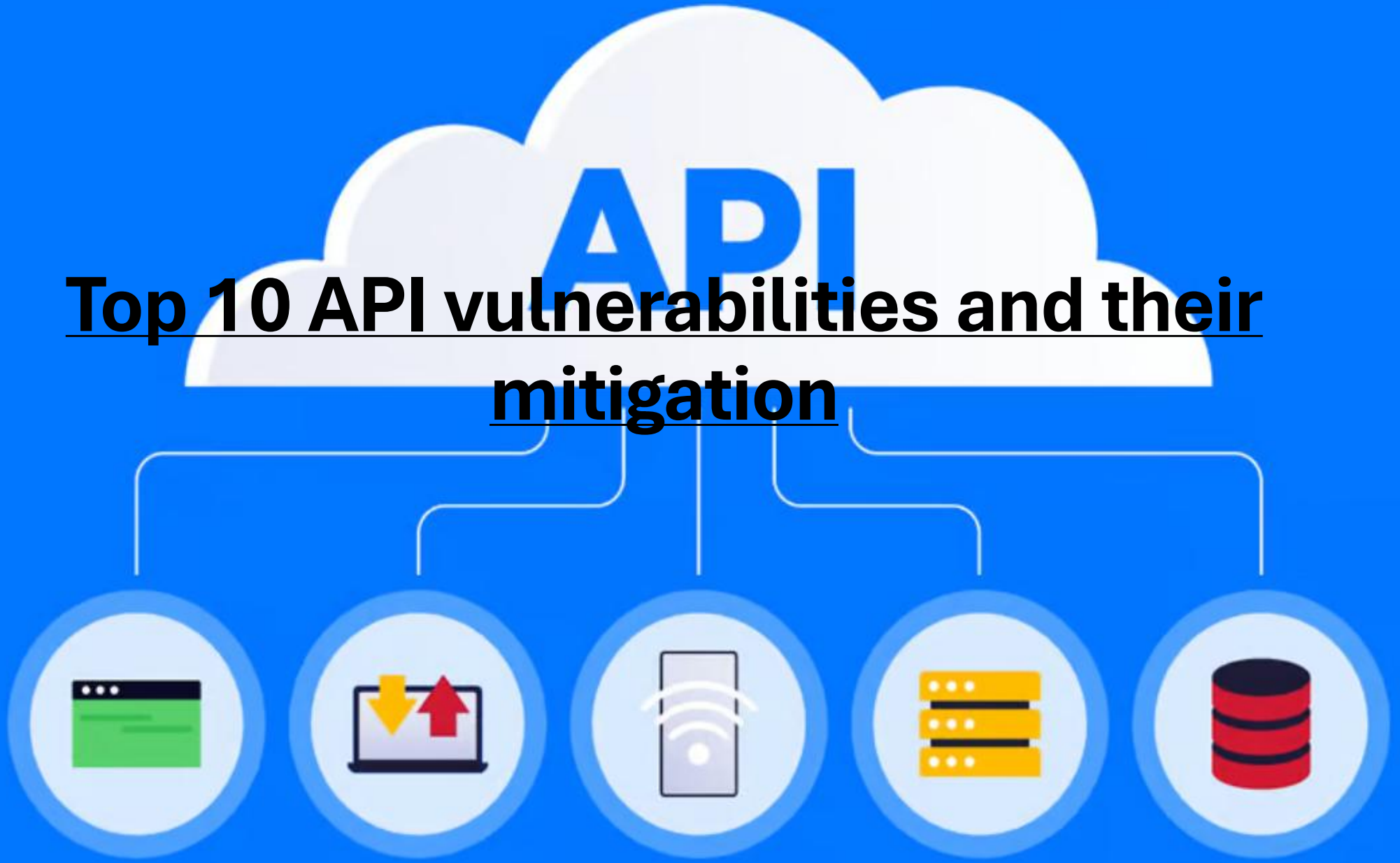




API Basics

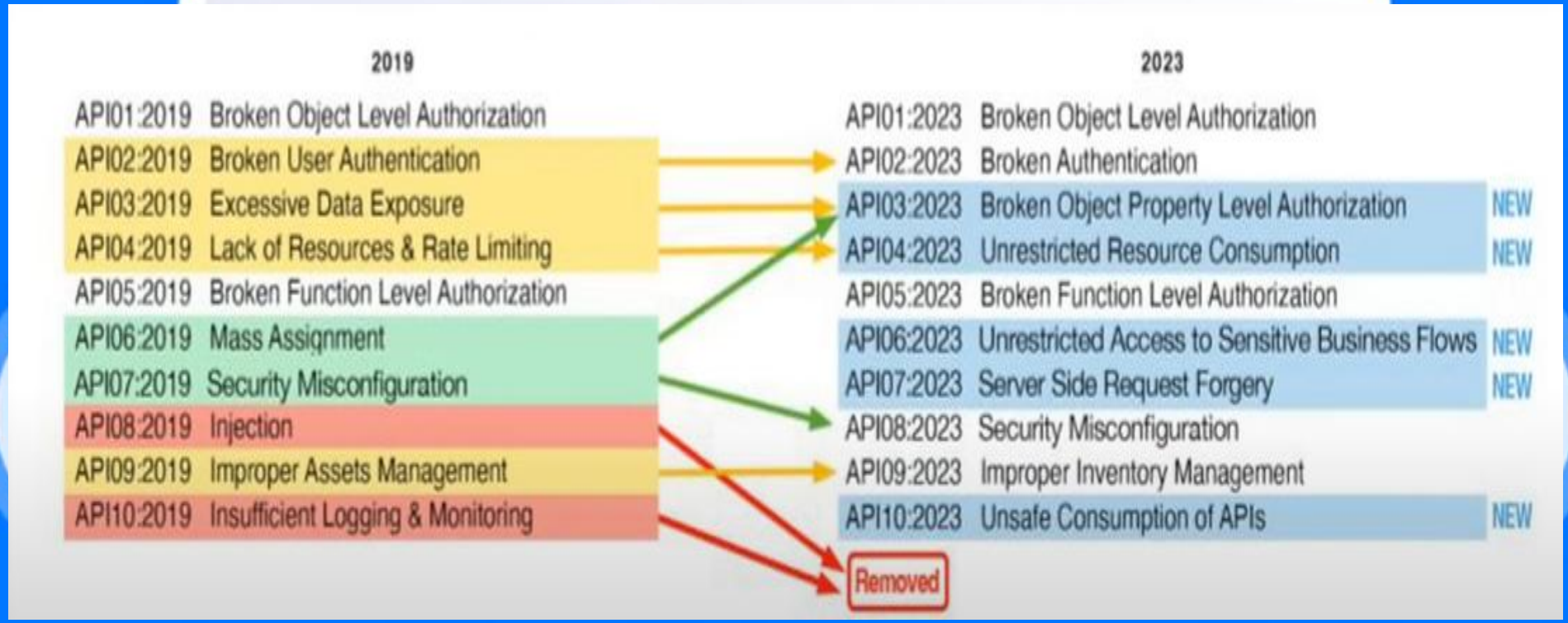
Nowadays a cloud-based solution like Web Application and API Protection (WAAP) becomes more and more popular that combines API gateway and a WAF with Machine Learning (ML) to do things like OpenAPI scheme auto discovery from the seen traffic, auto DOS rate limiting and blocking of malicious traffic and much more.

Top 10 API vulnerabilities and their mitigation



Top 10 API vulnerabilities and their mitigation

OWASP API 2019 vs OWASP API 2023. API are still vulnerable to injection attacks but as WAF products are now more advanced and block most of command or sql injection or other injection attacks.



Top 10 API vulnerabilities and their mitigation

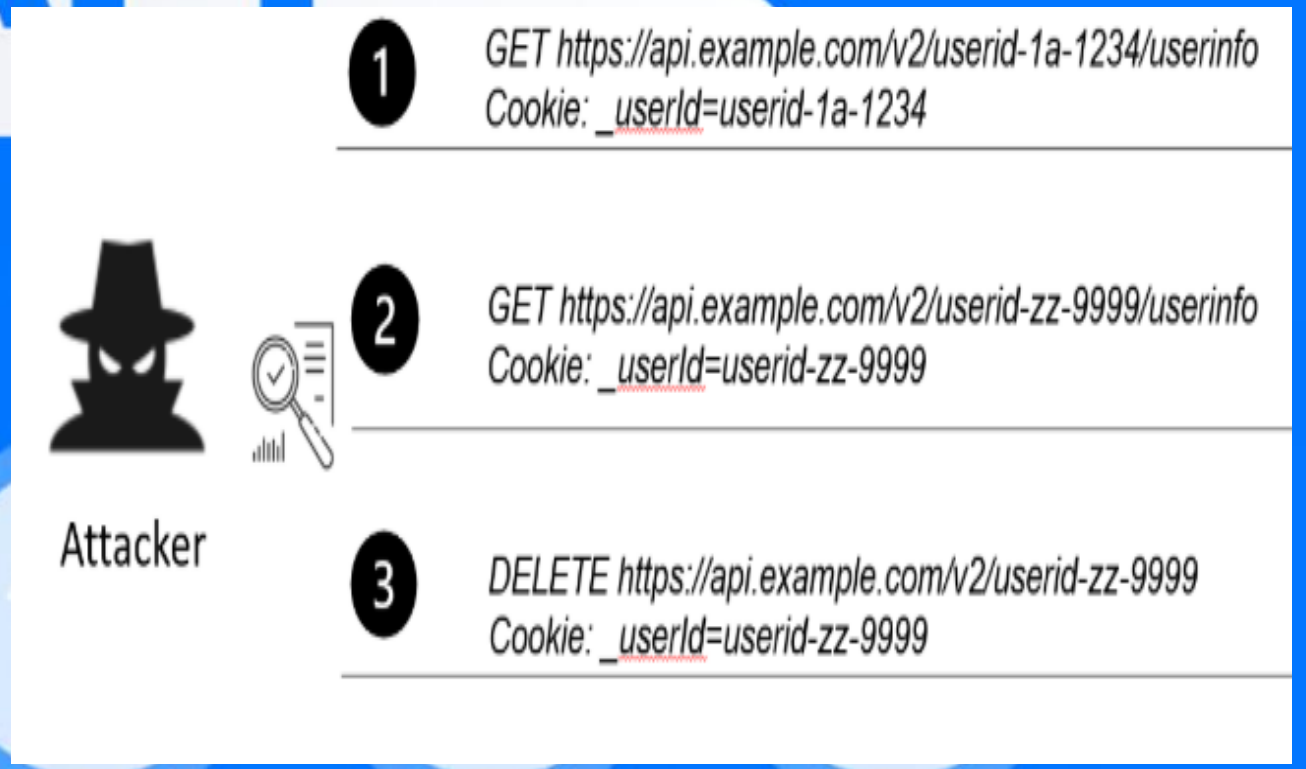
- Broken object level authorization (API1:2023)

Broken object level authorization occurs when authorization rules that dictate which object can be accessed by which users do not exist or are not applied properly. This allows attackers to manipulate API request parameters to access objects to which they are not authorized to access.

Top 10 API vulnerabilities and their mitigation

In the following attack scenario, the attacker manipulates the userid value in the URI path to retrieve user information from other users' account.

1. The attacker uses a legitimate account and sends the following API request to retrieve their user information.GET https://api.example.com/v2/userid-1a-1234/userinfo
   Cookie: _userId=userid-1a-1234
2. The attacker creates a second legitimate account, and using the userid, the attacker guesses and enumerates different userid values. They use a script or tool to send API requests, testing a range of userids, and receive sensitive information of other users.GET https://api.example.com/v2/userid-1a-1111/userinfo
   Cookie: _userId=userid-1a-1111
   GET https://api.example.com/v2/userid-zz-9999/userinfo
   Cookie: _userId=userid-zz-9999
3. They expand the scope of the attack by manipulating the API request to delete users.DELETE https://api.example.com/v2/userid-1a-1111
   Cookie: _userId=userid-zz-9999



Top 10 API vulnerabilities and their mitigation

- To protect your APIs, best practices recommend the following:
- implement authorization mechanisms to check if the logged-in user has access to perform the requested action on the record in every function that uses an input from the client to access a record in the database.
- Be extremely methodical and precise when testing the authorization mechanisms to ensure every API endpoint is secured against attacks on every object possible according to business requirements.
- Securing against broken object level authorization attacks is about more than just rolling out fixes at the source code level. It involves implementing a series of defenses at each layer in the authentication and authorization process.

The protections usually are implemented on the application server but this can be delegated to a API gateway device that can be integrated with a central IdP solution for authentication and authorization using federation services like OAuth, OpenID connect, JWT, and so on.

Top 10 API vulnerabilities and their mitigation

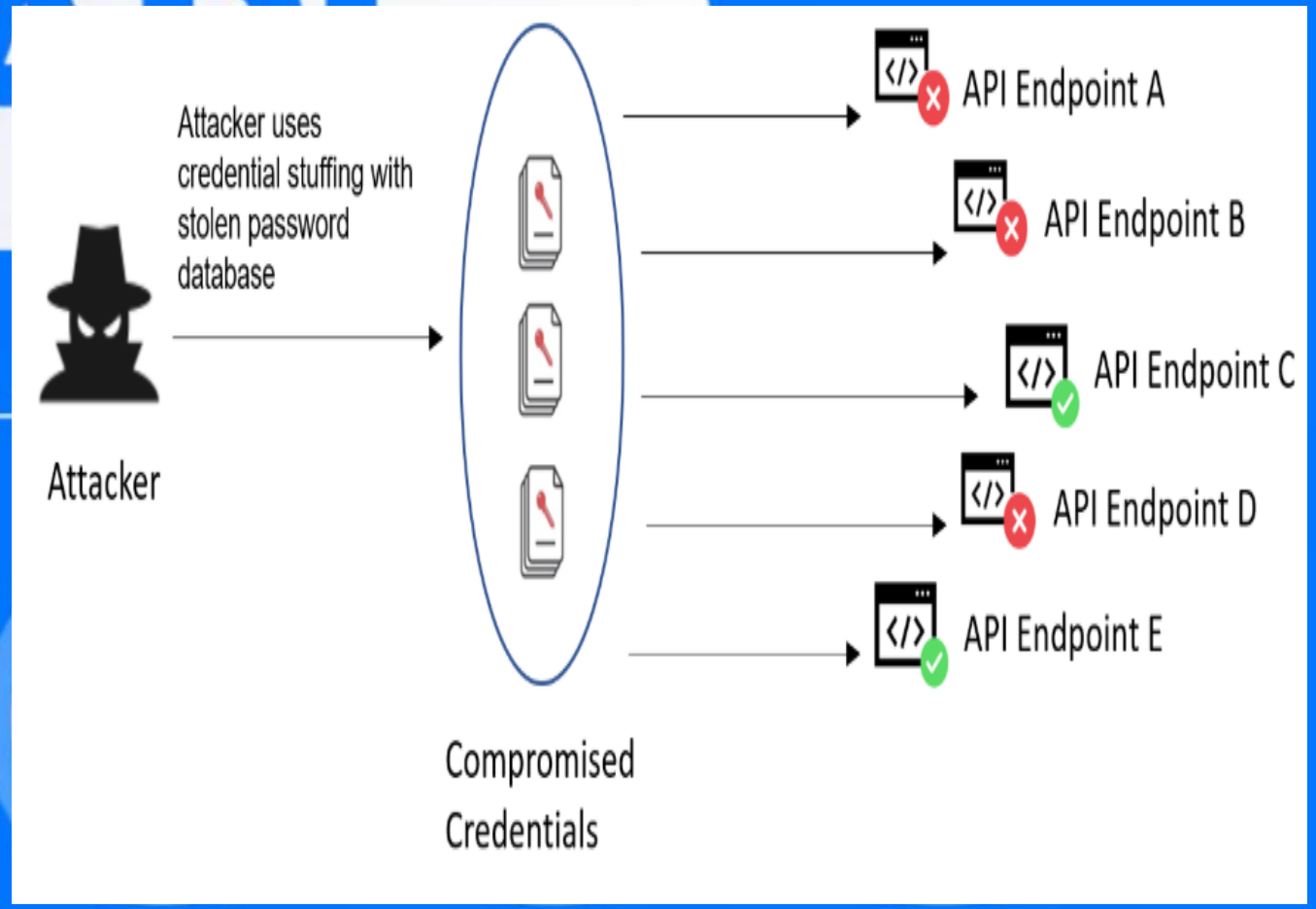
- Broken authentication (API2:2023)

Broken authentication occurs when the API authentication mechanisms are lacking or are not implemented correctly, allowing attackers to gain complete control of other users' accounts in the system, read their personal data, and perform sensitive actions on their behalf. Attacks that target authentication endpoints include brute-force, credential stuffing, and credential cracking.

Top 10 API vulnerabilities and their mitigation

In the following scenario, an attacker performs credential stuffing attacks against different API endpoints. Two authentication endpoints do not have proper authentication mechanisms implemented.

1. The attacker obtains a password database from a hacker forum.
2. Since a weak hashing algorithm was used to encrypt passwords, the attacker can expose the user credentials.
3. The attacker uses credential stuffing tools to test credential pairs on different API endpoints.
4. If the login is successful, the attacker knows they have a set of valid credentials.



Top 10 API vulnerabilities and their mitigation

To protect your APIs, best practices recommend that you use industry standard authentication mechanisms that include multi-factor, brute force protection, and account lockout to secure all API endpoints and authentication flows.

To secure your APIs against broken authentication attacks, you should use industry standard authentication methods. Your authentication mechanism should also be able to implement step-up authentication when the conditions or criteria for authentication changes. Examples of good authentication options are:

- Multi-factor authentication through RADIUS authentication
- Certificate-based authentication
- Password-based authentication like LDAP, HTTP, Radius, Active Directory and so on.

Top 10 API vulnerabilities and their mitigation

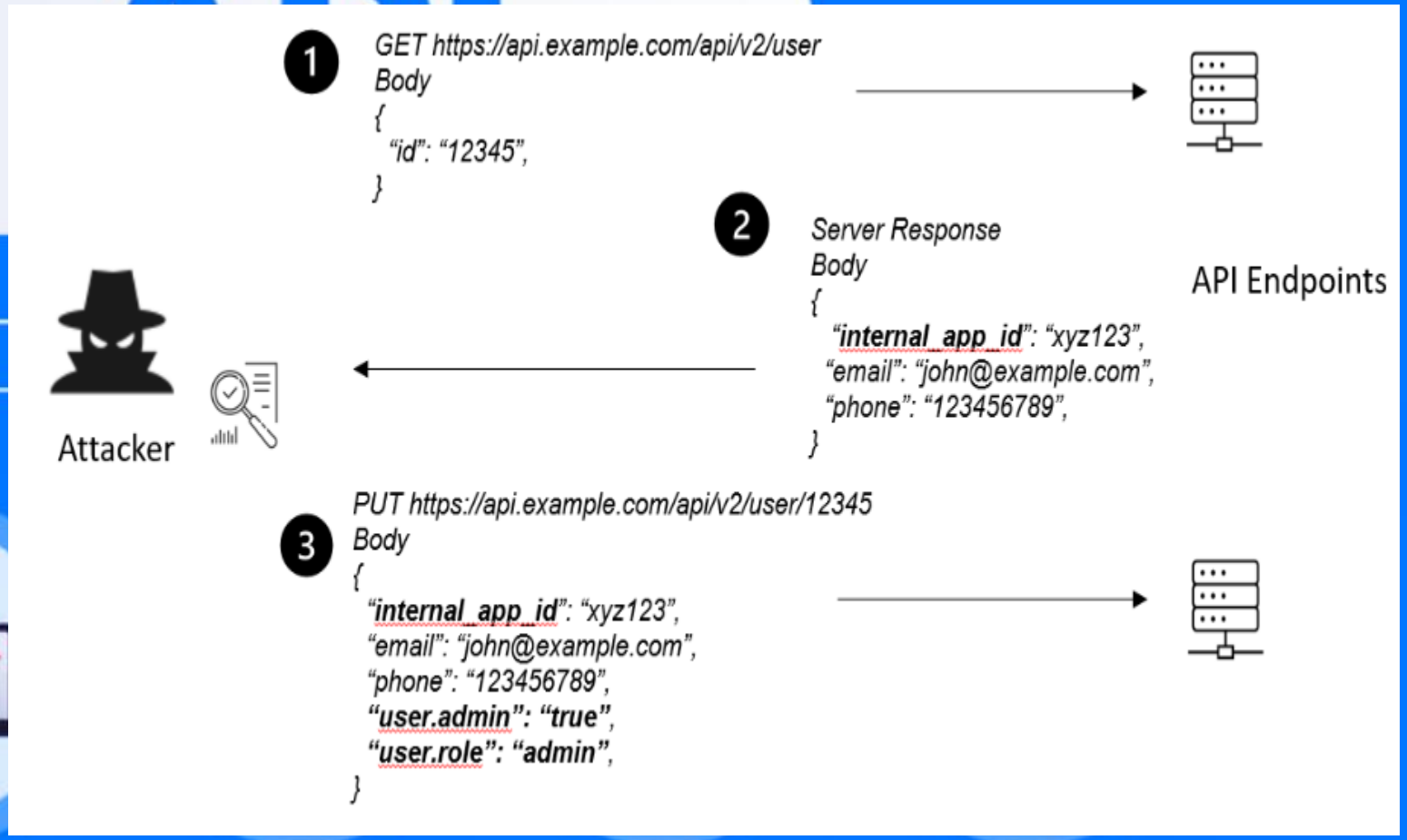
- Broken object property level authorization (API3:2023)

Broken object property level authorization occurs when the API is designed to return all properties of an object without considering the sensitivities of each property. This leads to the API returning more data than required by the end user, thereby increasing the risk of exposure of sensitive data.

Top 10 API vulnerabilities and their mitigation

In the following scenario, an attacker exploits excessive data exposure from the API server to obtain an internal application ID and uses it to illegally escalate their own account privileges:

1. An attacker sends a legitimate API request to request their account details.
2. The API server, relying on the front end to filter sensitive data, returns all properties of the account object, including an internal application ID which is applicable only for administrator accounts.
3. The attacker adds the **user.admin**, **user.role** parameter s to the JSON user object in the API request.



Top 10 API vulnerabilities and their mitigation

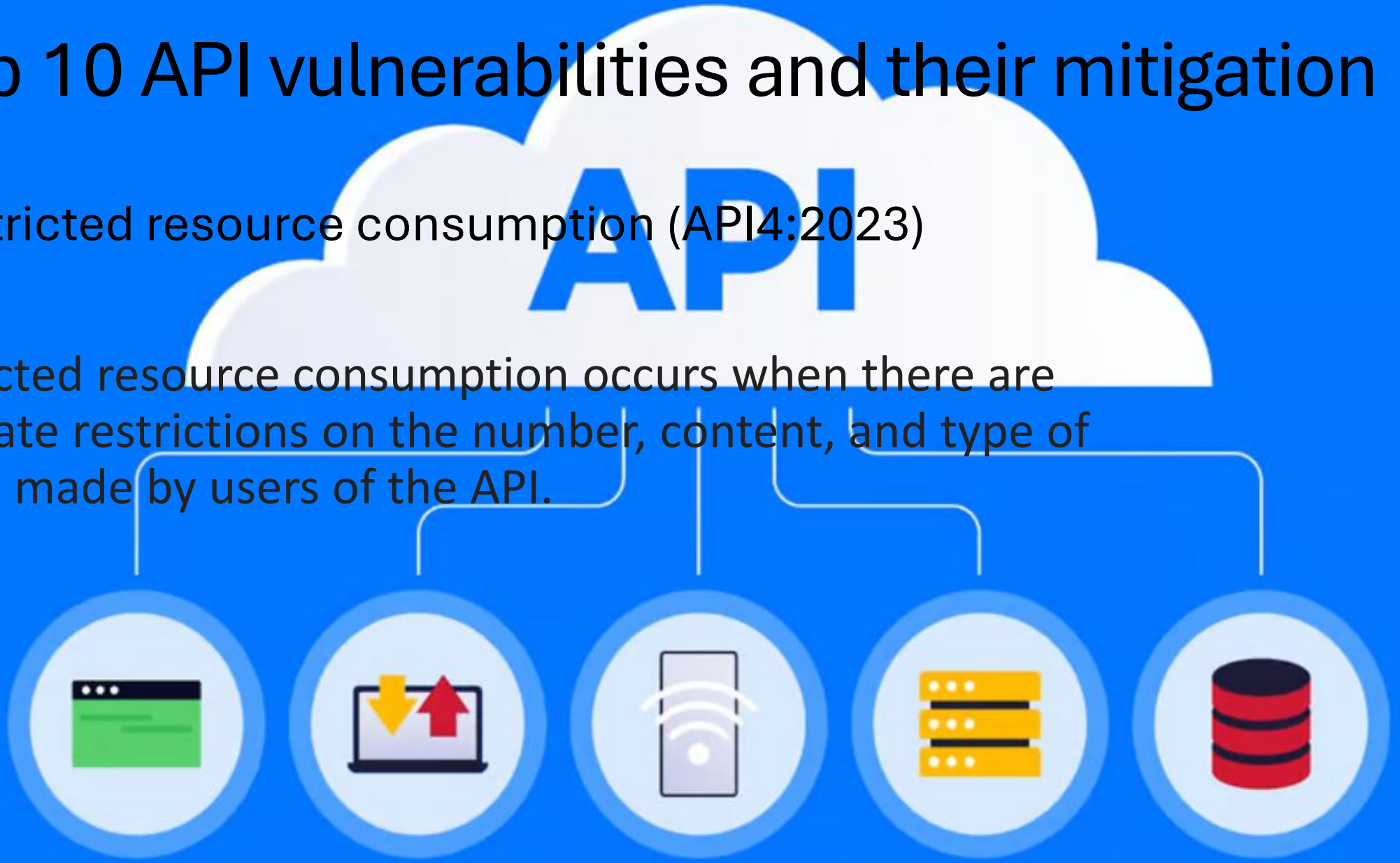
- Most traditional security tools lack the capability to identify which data is sensitive from the legitimate data returned by the API. Some tools may be able to detect and mask or block well-defined sensitive data types, such as credit card numbers (CCNs) or social security numbers (SSNs). When designing an API, best practices recommend that you never rely on the client side to filter sensitive data and always review the responses directly from the API to make sure they contain only legitimate, required data. For example, ensure that the API returns specific properties that are required instead of all properties.
- In addition, to prevent attackers from illegally modifying sensitive object properties, best practices recommend that you define a clear and comprehensive allow/deny list of all parameters, objects, and properties and examine and validate all client inputs against the list to ensure all client requests conform to your application schema.

WAF systems usually can block or mask responses that have data matching patterns of known sensitive data, like limited Data Loss prevention (DLP) systems. Also, Openapi spec can document what is accepted in JSON or XML requests and API Gateway with ingested API specification can enforce the OpenAPI spec.

Top 10 API vulnerabilities and their mitigation

- Unrestricted resource consumption (API4:2023)

Unrestricted resource consumption occurs when there are inadequate restrictions on the number, content, and type of requests made by users of the API.



Top 10 API vulnerabilities and their mitigation

In the following scenario an attacker manipulates the content of the API request, leading to a server response with an excessively large payload, resulting in slow application performance.

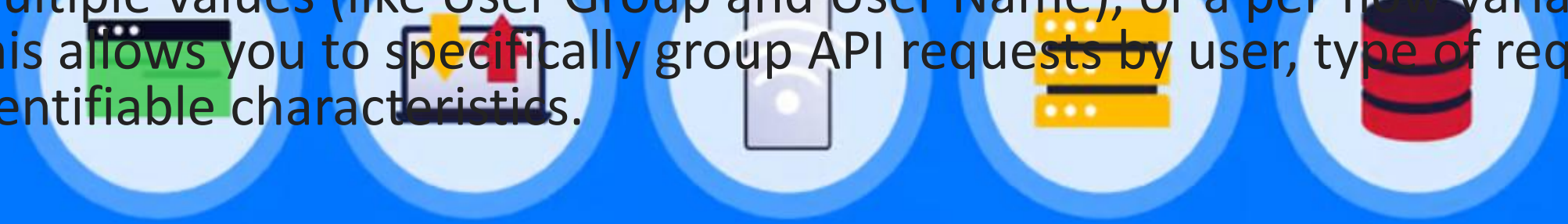
1. An attacker sends a legitimate API request for a single page with 200 items.
2. The attacker modifies the page size, increasing it from 200 to 200,000.

This increase causes the application to return an excessive number of items in response to the query, consuming excessive server resources.

Top 10 API vulnerabilities and their mitigation

To protect your APIs, best practices recommend that you implement limits (CPU, memory, processes, and so on) on how clients can call your APIs and examine and validate client queries to ensure query strings and request that body parameters conform to your application schema, maximum size, and other requirements.

API Gateways can enforce strong rate limits based not only on ip addresses but also based on entities such as Client ID, User Group, Client IP address, User Name, multiple values (like User Group and User Name), or a per-flow variable name. This allows you to specifically group API requests by user, type of request, or any identifiable characteristics.



Top 10 API vulnerabilities and their mitigation

- Broken function level authorization (API5:2023)

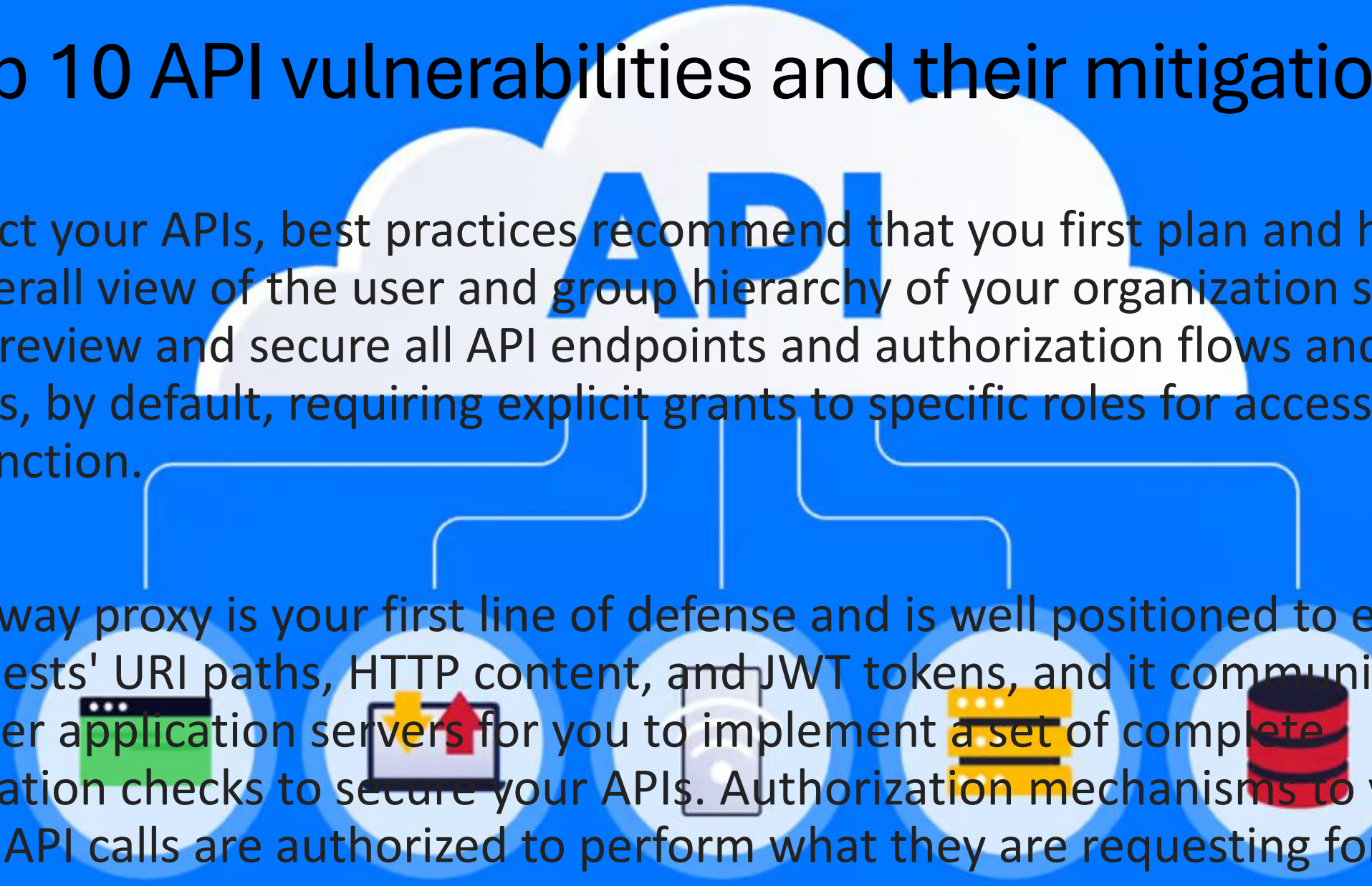
Broken function level authorization occurs when API user functions are not clearly defined or are too complex and, as a result, authorization checks are not completely implemented based on clear user group and roles.

Top 10 API vulnerabilities and their mitigation

In the following scenario, the attacker buys all the stock of a high-demand product and resells it at a higher price:

1. An attacker uses code to automatically buy the next-generation release of a popular new shoe.
2. Since the API doesn't implement the appropriate protection, the attacker uses a bot to run the code and buy up the entire stock before other legitimate users. The attacker later resells the shoes at a profit.

Top 10 API vulnerabilities and their mitigation



To protect your APIs, best practices recommend that you first plan and have a good overall view of the user and group hierarchy of your organization so that you can review and secure all API endpoints and authorization flows and deny all access, by default, requiring explicit grants to specific roles for access to every function.

API gateway proxy is your first line of defense and is well positioned to examine API requests' URI paths, HTTP content, and JWT tokens, and it communicates with other application servers for you to implement a set of complete authorization checks to secure your APIs. Authorization mechanisms to validate that the API calls are authorized to perform what they are requesting for.

Top 10 API vulnerabilities and their mitigation

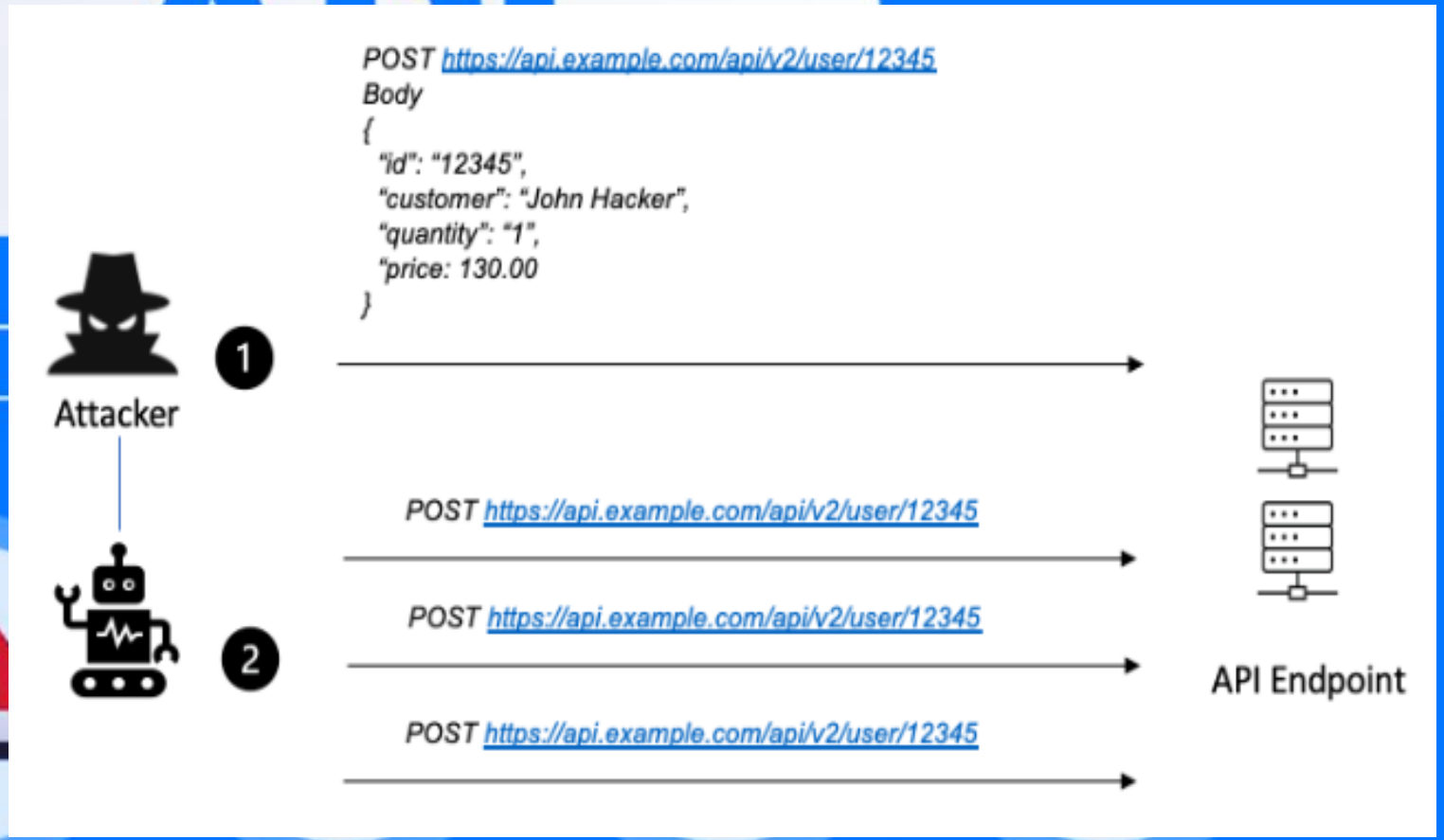
- Unrestricted access to sensitive business flows (API6:2023)

Unrestricted access to sensitive business flows occurs when an API exposes a business flow without regard to whether excessive access to the flow may harm the business.

Top 10 API vulnerabilities and their mitigation

In the following scenario, the attacker buys all the stock of a high-demand product and resells it at a higher price:

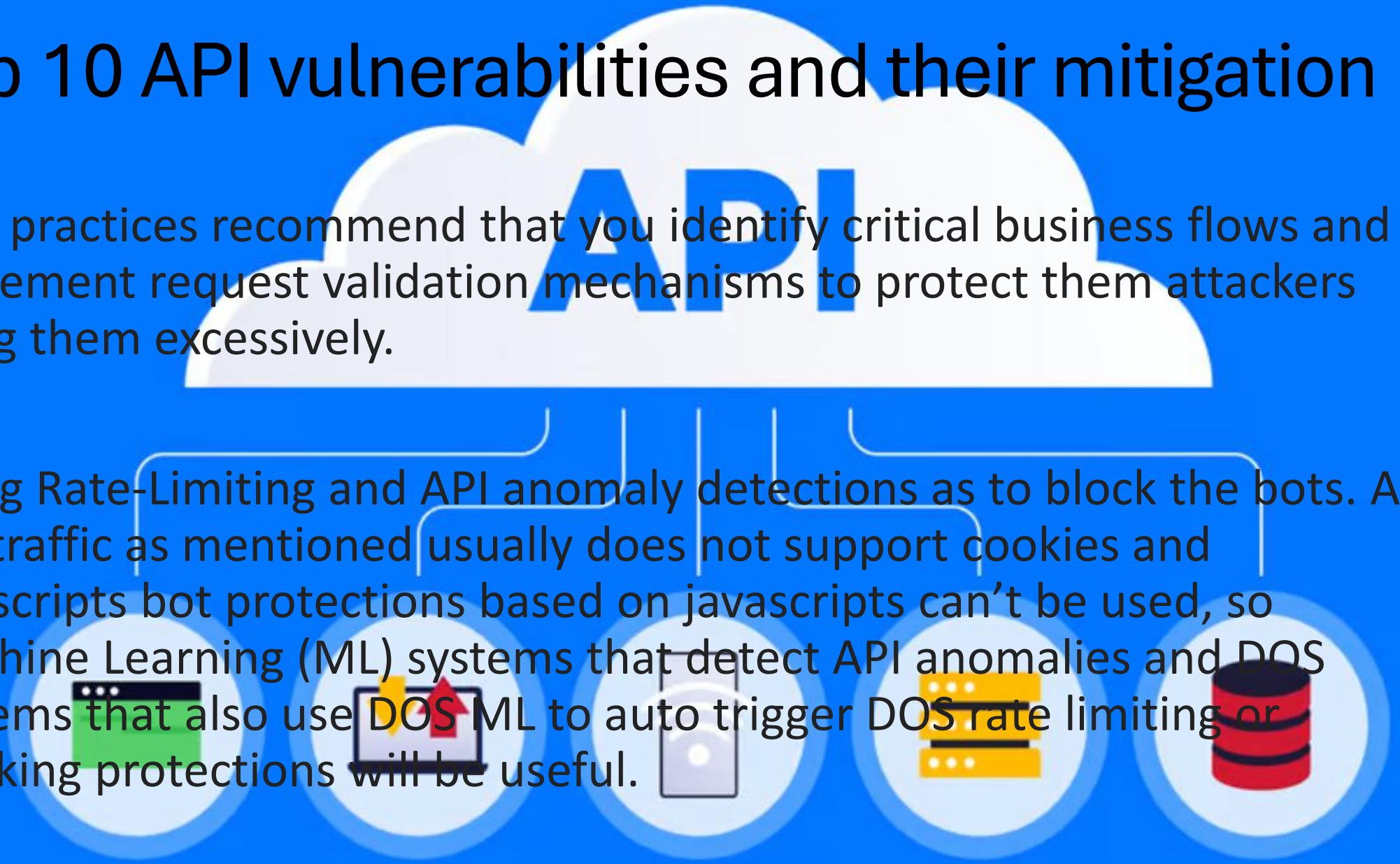
1. An attacker uses code to automatically buy the next-generation release of a popular new shoe.
2. Since the API doesn't implement the appropriate protection, the attacker uses a bot to run the code and buy up the entire stock before other legitimate users. The attacker later resells the shoes at a profit.



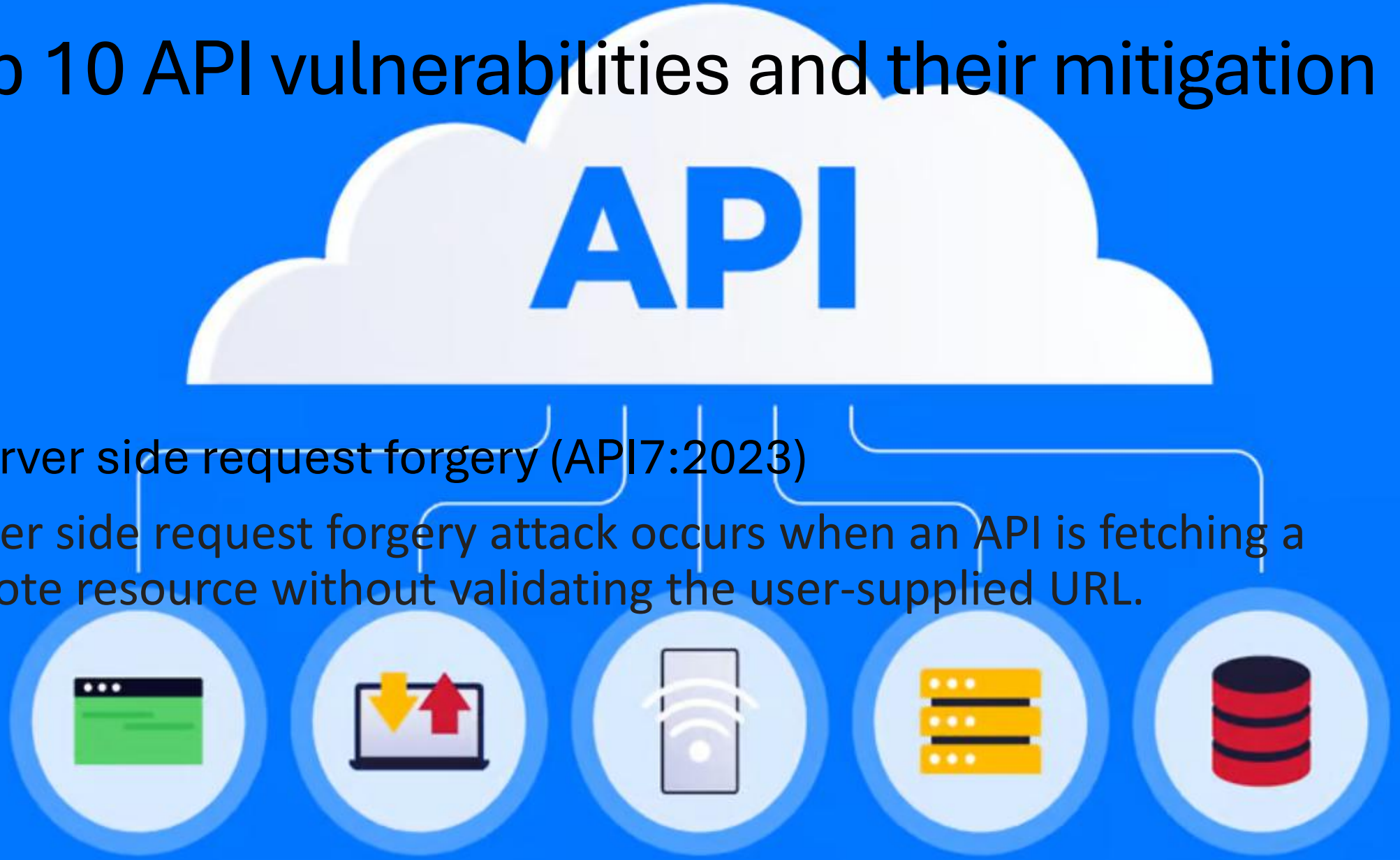
Top 10 API vulnerabilities and their mitigation

Best practices recommend that you identify critical business flows and implement request validation mechanisms to protect them attackers using them excessively.

Using Rate-Limiting and API anomaly detections as to block the bots. As API traffic as mentioned usually does not support cookies and javascripts bot protections based on javascripts can’t be used, so Machine Learning (ML) systems that detect API anomalies and DOS systems that also use DOS ML to auto trigger DOS rate limiting or blocking protections will be useful.



Top 10 API vulnerabilities and their mitigation



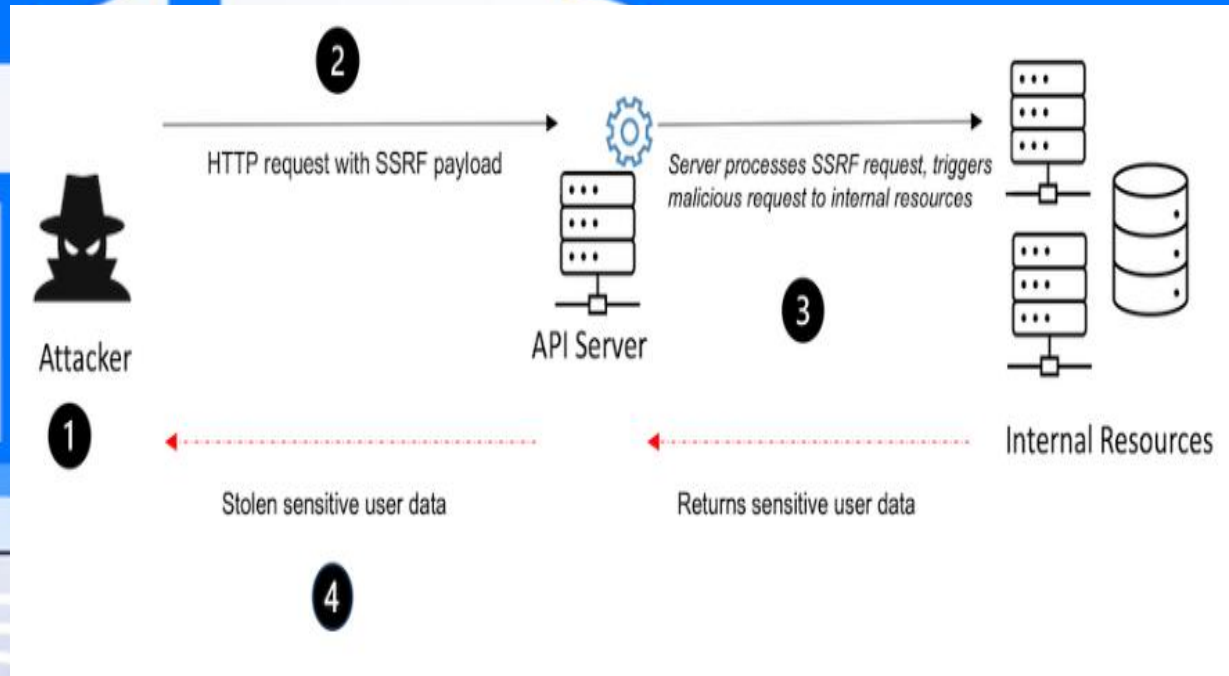
- Server side request forgery (API7:2023)

Server side request forgery attack occurs when an API is fetching a remote resource without validating the user-supplied URL.

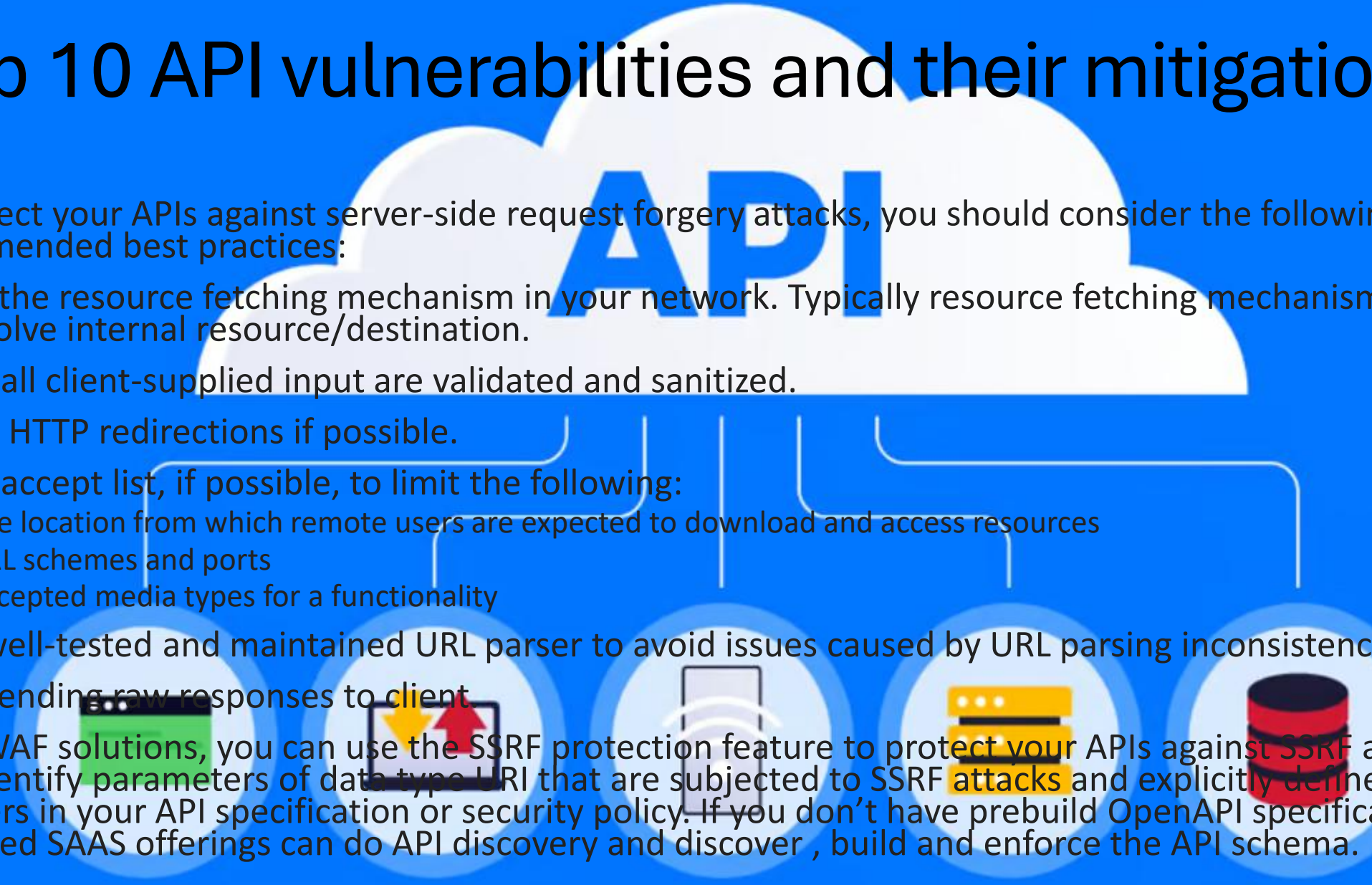
Top 10 API vulnerabilities and their mitigation

In the following attack scenario, an attacker exploits an API that makes calls to an internal resource on the same network.

1. The attacker identifies an API that is vulnerable to SSRF attacks.
2. The attacker sends a forged request to the vulnerable API and targets the internal resource that resides on the same network.
3. The API sends the forged request to the internal resource and receives a response with the requested data.
4. The API sends the response data back to the attacker, bypassing detection.



Top 10 API vulnerabilities and their mitigation



- To protect your APIs against server-side request forgery attacks, you should consider the following recommended best practices:
- Isolate the resource fetching mechanism in your network. Typically resource fetching mechanism should not involve internal resource/destination.
- Ensure all client-supplied input are validated and sanitized.
- Disable HTTP redirections if possible.
- Use an accept list, if possible, to limit the following:
  - The location from which remote users are expected to download and access resources
  - URL schemes and ports
  - Accepted media types for a functionality
- Use a well-tested and maintained URL parser to avoid issues caused by URL parsing inconsistencies.
- Avoid sending raw responses to client.

In most WAF solutions, you can use the SSRF protection feature to protect your APIs against SSRF attacks. You should identify parameters of data type URI that are subjected to SSRF attacks and explicitly define the URI parameters in your API specification or security policy. If you don't have prebuild OpenAPI specification the cloud based SAAS offerings can do API discovery and discover , build and enforce the API schema.

Top 10 API vulnerabilities and their mitigation

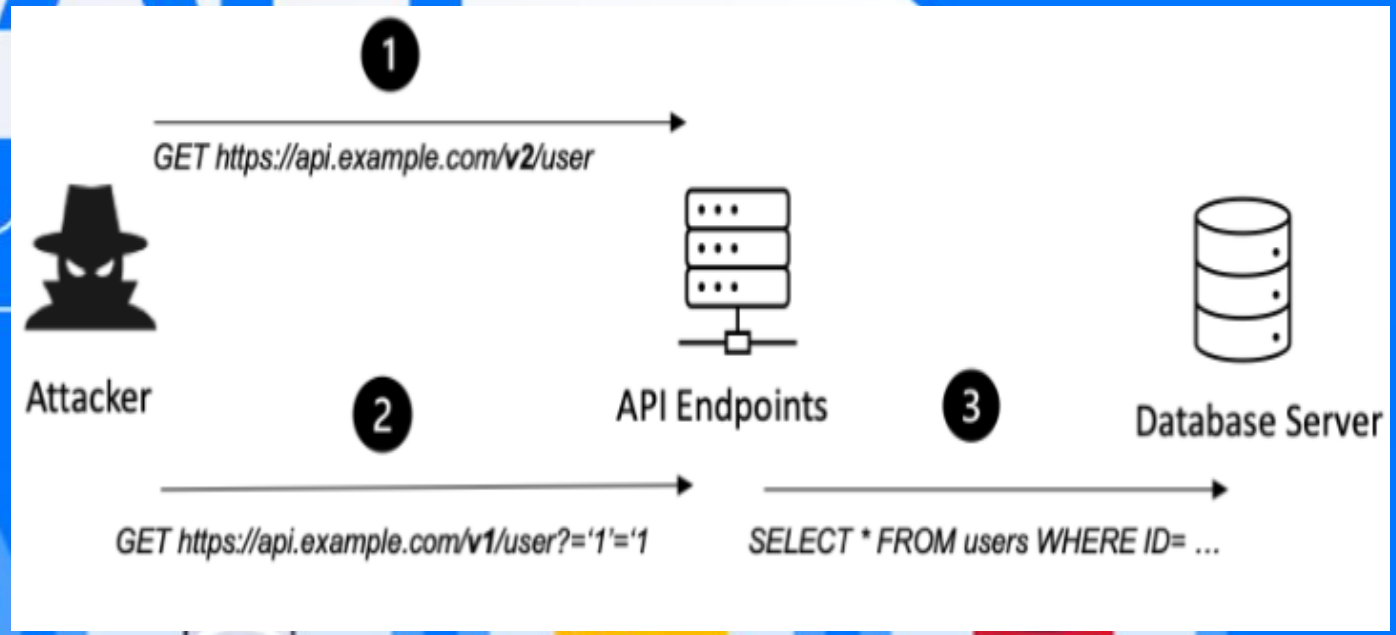
- Improper inventory management (API9:2023)

Improper inventory management vulnerabilities can occur when security best practices are not followed during the API development cycle. Not following security best practices in the design phase may lead to vulnerabilities such as old API versions, unpatched systems, outdated API documentation, and unnecessarily exposed API endpoints/

Top 10 API vulnerabilities and their mitigation

In the following scenario the attacker finds old, unpatched API endpoints that are still connected to the production database.

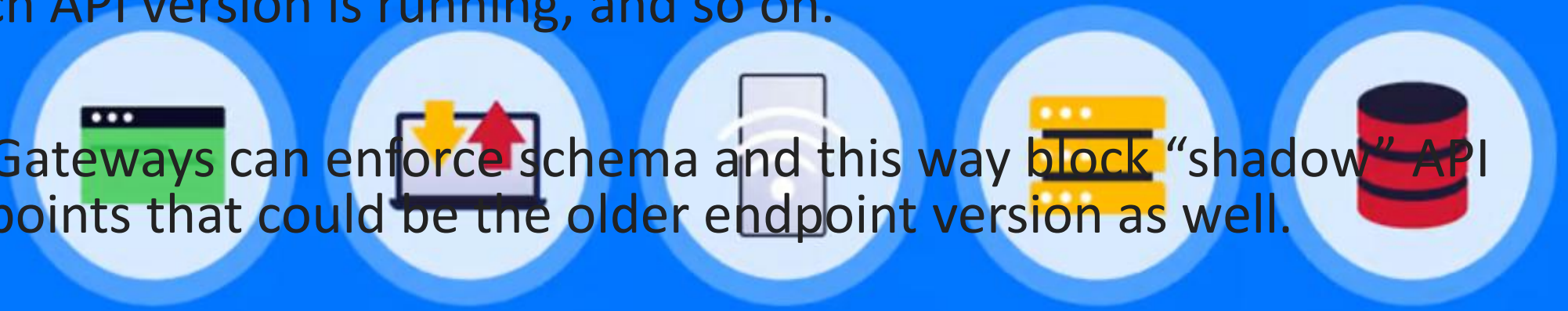
1. The attacker uses an automated scanning tool to search production APIs on the target system.
2. The scanning tool finds old, unprotected API resources.
3. Since the old API is not running the latest code, the attacker uses injection attacks to compromise the production database.



Top 10 API vulnerabilities and their mitigation

To protect your APIs, best practices recommend that you implement a repeatable process that is audited for security when setting up and configuring your environment. Include, in your configurations, the F5 features listed in the following tables.

You should have a simple, clear, and repeatable process when updating to new API versions. This process makes it clear which environment the API is running, who has access to the production and test servers, which API version is running, and so on.

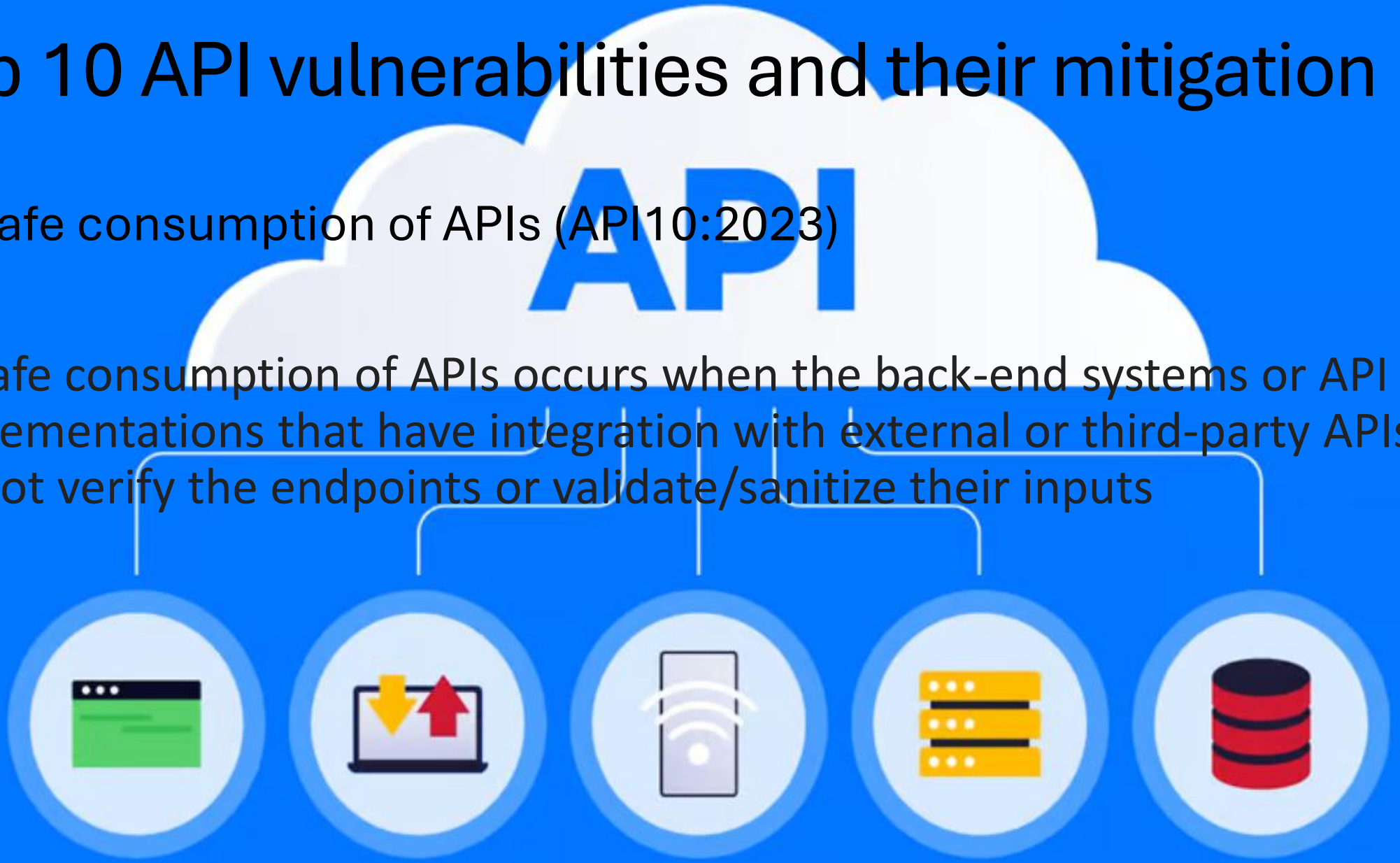


API Gateways can enforce schema and this way block “shadow” API endpoints that could be the older endpoint version as well.

Top 10 API vulnerabilities and their mitigation

Unsafe consumption of APIs (API10:2023)

Unsafe consumption of APIs occurs when the back-end systems or API implementations that have integration with external or third-party APIs do not verify the endpoints or validate/sanitize their inputs

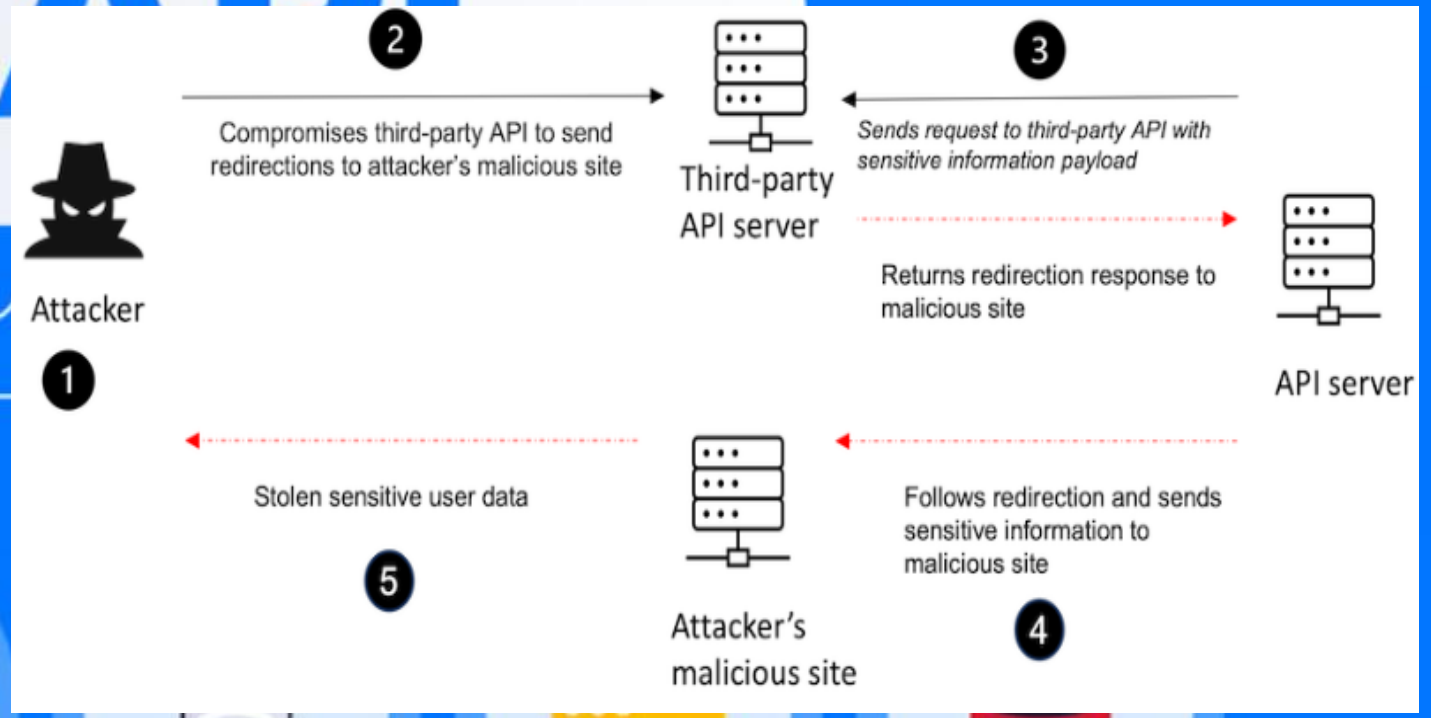


Top 10 API vulnerabilities and their mitigation

In the following attack scenario, an attacker exploits a third-party API that your back-end system or API implementation is integrated with.

1. The attacker identifies a third-party API that is vulnerable and compromises it.
2. The attacker uses the third-party API to send redirections to the attacker's malicious site.
3. Your back-end system or API implementation sends a request to the third-party API with payload that contains sensitive information and receives a redirection response from the third-party API.
4. Your backend system or API implementation then follows the third-party API's redirection without validation and re-sends the request with the sensitive information to the attacker's malicious site.
5. The attacker now has a copy of the sensitive information.

•



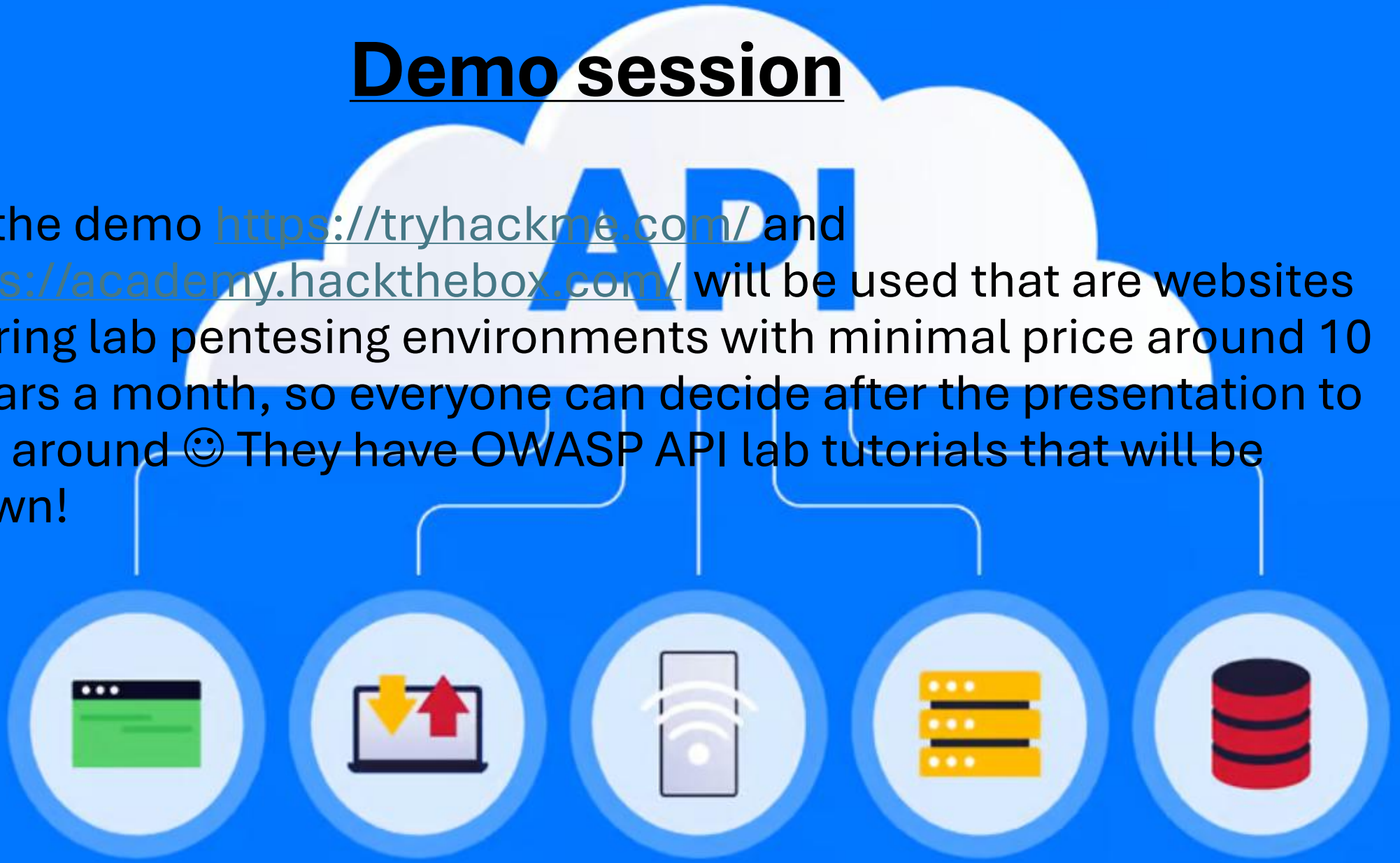
Top 10 API vulnerabilities and their mitigation

- To protect your APIs against unsafe consumption of APIs attacks, you should consider the following recommended best practices:
- Evaluate the security posture and practices of the external or third-party API provider which your back-end system or API implementation is integrated with.
- Ensure all data received from the external or third-party APIs is validated and sanitized before using it.
- Use an allowlist, if possible, to limit where the external or third-party API provider may redirect. If possible, do not follow redirection without validation.
- Implement timeouts when interacting with the external or third-party APIs.
- Limit resources when processing the responses from external or third-party APIs.

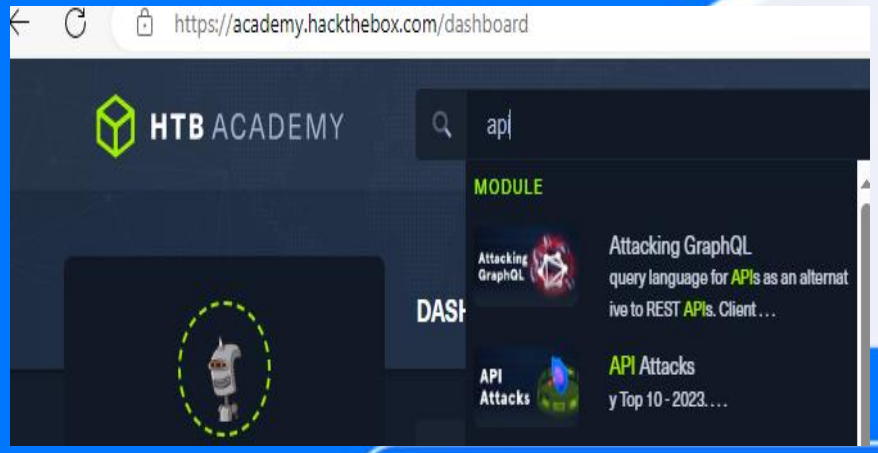
The primary weakness in unsafe API consumption comes from the interaction with third-party API servers whose security posture could be beyond your control. Best practice therefore recommends that you implement a zero trust environment to differentiate these third-party APIs from your local APIs in the authentication, authorization, and validation workflow.

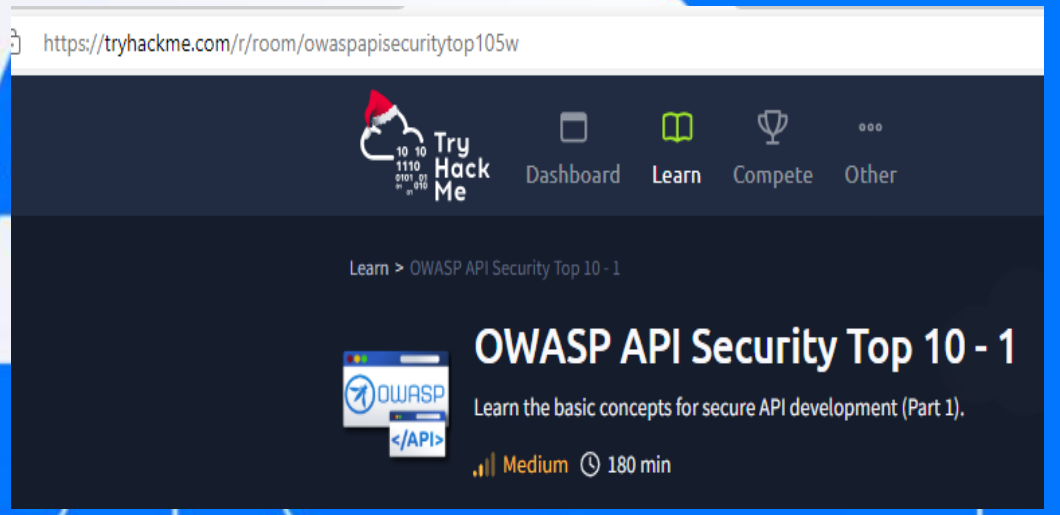
Demo session

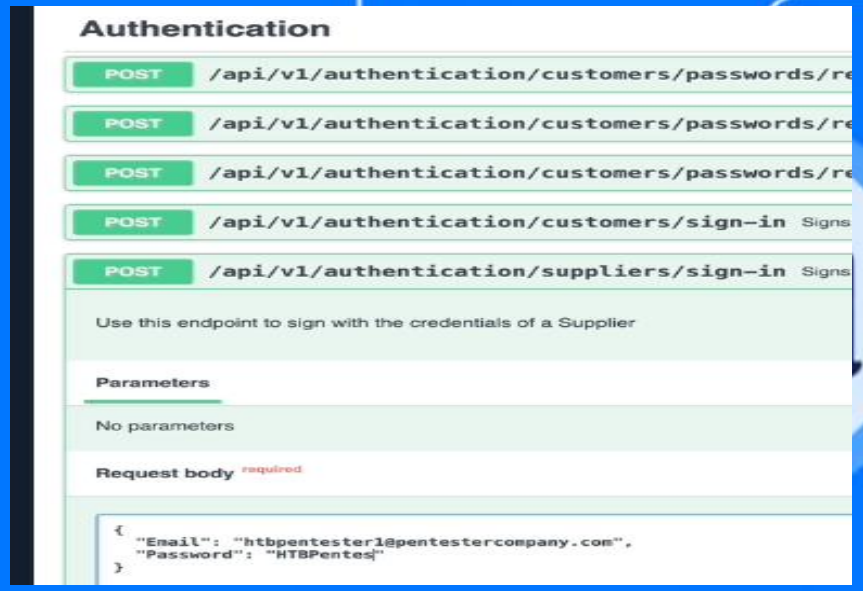
For the demo https://tryhackme.com/ and https://academy.hackthebox.com/ will be used that are websites offering lab pentesing environments with minimal price around 10 dollars a month, so everyone can decide after the presentation to play around ☺ They have OWASP API lab tutorials that will be shown!

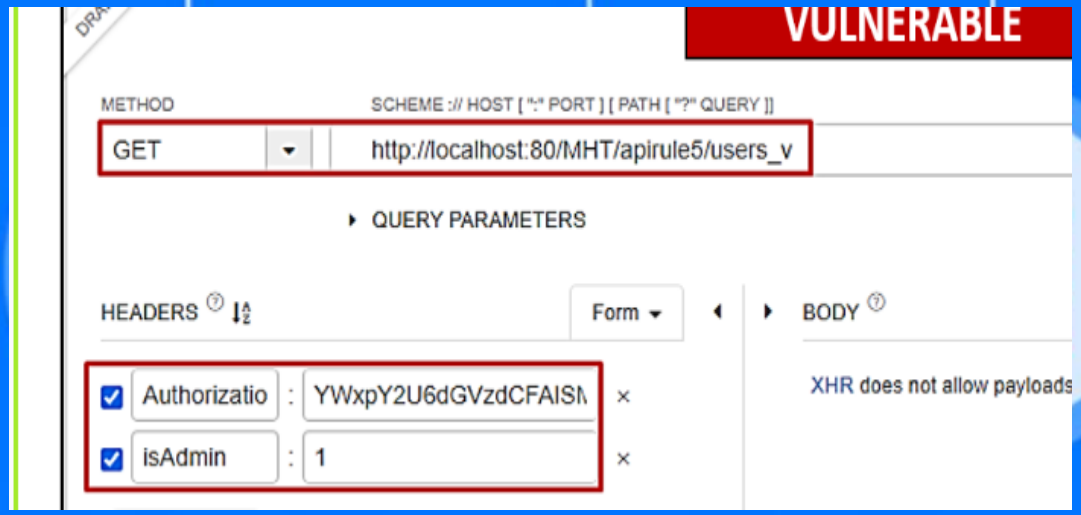


Demo session









Injections

Injection attacks even if not in OWASP 2023 can still be an issue like sql injections, xxe XML injection or json nosql injections.

Extra useful links:

OWASP Top 10 API Security Risks – 2023 - OWASP API Security Top 10

OWASP API Security Top 10 Overview & Best Practices | F5

Article Detail

OWASP API Security Top 10 Course – Secure Your Web Apps

What Is OpenAPI? | Swagger Docs

Swagger UI

API Security Fundamentals Course

Contacts:

(18) Nikolay Dimitrov | LinkedIn

THE END!