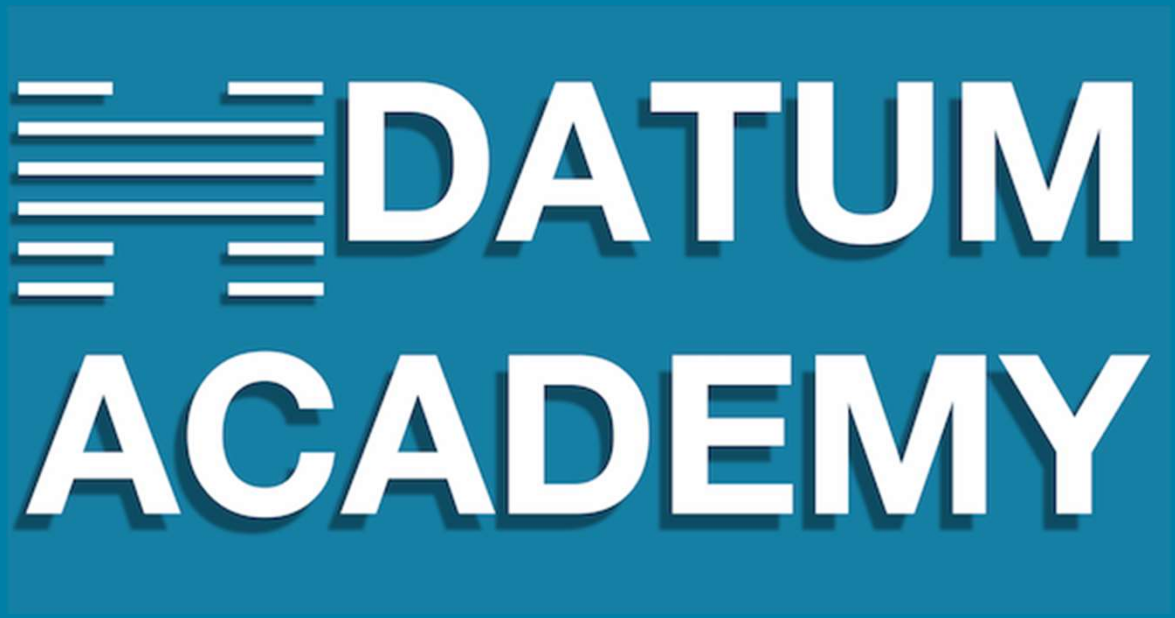


Web Access Control Strategies

Pr. Alban Gabillon
Université de la Polynésie Française

Outline

- Security Policy
- Access Control Models
- Resource sharing using a Cross-Origin Resource Sharing (CORS) policy
- Access Delegation using the OAuth 2.0 protocol
- Identity and Access Management (IAM) policies in the cloud

Security Policy

- Cybersecurity is the protection of the information system against malicious actions
- The security policy consists of a set of rules defining what is permitted and what is not
- In the simplest case, each security rule is a permission (authorization)
  - The default Policy is closed i.e., what is not allowed is forbidden
- A system is secure if and only if the security policy cannot be violated
- Securing a system (w.r.t to a security policy) requires security mechanisms
  - Authentication mechanisms
  - Access control mechanisms
  - Information flow mechanisms
  - Encryption mechanisms
  - Etc.

Security policy

- In addition to permissions, a security policy may also include explicit prohibitions
- In order to specify exceptions to general permissions
  - Children are permitted to play video games except for the youngest sister
- Some regulations are easier to express through prohibitions
  - Video forbidden to under 18

Security policy

- Some regulations may also include obligations
  - Doctors have the obligation to retain old medical records (against possible future need)
- There are relationships between permission/prohibition/obligation:
  - The obligation to do corresponds to the prohibition not to do
    - Doctors are prohibited from not retaining (i.e. deleting) old medical records
  - The non obligation to do corresponds to the permission not to do
  - Permission to do corresponds to the non-prohibition to do

Security policy

- A Permission, prohibition or obligation may also depend on one or several conditions leading to the concept of dynamic security rules
  - Temporal condition
    - Students can use the Wi-Fi during the opening hours of the university
  - Spatial condition
    - Students are not allowed to connect to their account from outside the university
  - Provisional condition
    - After reading the disclaimer, users can install the software package
    - After obtaining your consent Google may use your location data
  - Etc.

Security policy

- A better definition of security policy
  - A security policy is a list of security rules where each rule is either a permission, a prohibition or an obligation depending possibly on certain conditions
- Security Policies that mix permissions and prohibitions should include a conflict resolution policy
  - E.g., prohibition prevails

Access Controls Models

- Discretionary Access Control Model (DAC)
- Mandatory Access Control Model (MAC)
- Rôle Based Access Control Model (RBAC)
- Attribute Based Acces Control Model (ABAC)

Access Controls Models / DAC

- The Discretionary Access Control (DAC) policy is a policy where the security rules (permissions) are based on the identity of the subject (users)
- Its administration relies on the concept of ownership
  - Users define the security policy protecting their own objects (resources)
- Examples of systems implementing DAC are,
  - Windows / Unix,
  - Oracle SQL

Access Controls Models / DAC

- A DAC policy can be modelled by an access control matrix
  - Lines represent subjects
  - Columns represent objects
  - A Cell (intersection of a row and a column) contains all the actions that the subject has the right to perform on the object

	Code	Data
Alice	read,write,execute	read, write
Bob	read, execute	read

Implementation of DAC policy

- Representing the access control matrix as such in memory is obviously not an option
  - Matrix is huge and very sparse
- There are two solutions to represent the access control matrix within the computer system:
  - An access control list encodes the non-empty cells associated with a column (object)
  - A list of capabilities encodes the non-empty cells associated with a row (subject)

Implementation of DAC policy

DATUM ACADEMY

ACLs

<Code,(Alice,(r,w,x)),(Bob,(r,x))>

<Data,(Alice,(r,w)),(Bob,(r))>

Capabilities

<Alice,(Code,(r,w,x)),(Data,(r,w))>

<Bob,(Code,(r,x)),(Data,(r))>

Access Control List

		Code	Data
Capability List	Alice	read,write,execute	read, write
	Bob	read, execute	read

Access Controls Models / MAC

- The Mandatory Access Control (MAC) policy is a policy where the security rules are mandatory and cannot be modified even by a security administrator
- Examples of systems implementing MAC are,
  - Browser (Same origin policy),
  - Military applications (Multilevel security policy)

Access Controls Models / RBAC

- The Role Based Access Control (RBAC) policy is a security policy in which security rules (permissions) are based on user roles within an organization
- Security rules do not refer to users but to roles
- Roles are then assigned to users according to their function in the organization
- Roles can be organized in an inheritance hierarchy
- Examples of systems implementing RBAC are,
  - Teams Platform,
  - Oracle SQL,
  - Identity and Access Management (IAM)

Access Controls Models / ABAC

- The Attribute based Access Control (ABAC) policy is a security policy in which security rules (permissions, prohibitions and obligations) are conditional.
- Security rules are based on attributes relating to users, protected data and the environment
- The ABAC policy has high expressive power and consists of rules which can be represented by IF.. THEN .. Statements
  - Axiomatics (https://www.axiomatics.com/) offers a large suite of systems/security components implementing the ABAC policy

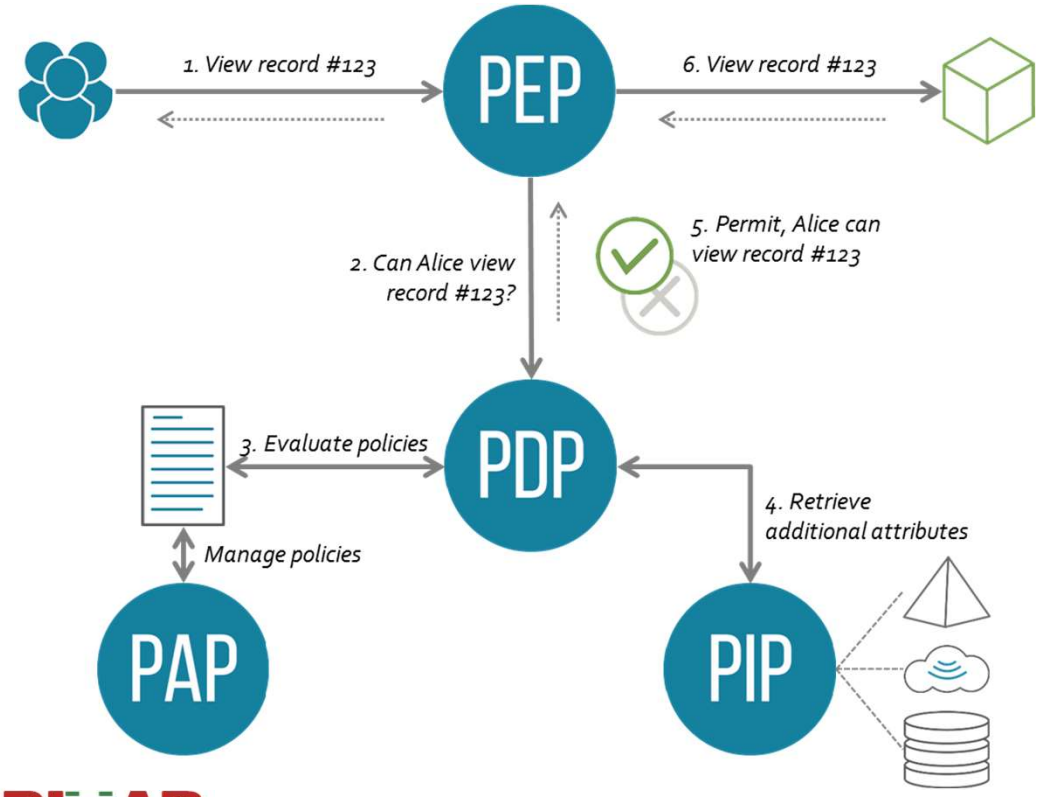
Access Controls Models / ABAC

- Examples of rules
- Students are permitted to use Wi-Fi between 7:00 and 19:00
  - $role(s, student) \wedge type(o, wifi) \wedge time < 7 \wedge time > 19 \rightarrow permission(s, use, o)$
- One may use the Wi-Fi only from inside the building
  - $gpsPosition(s, p) \wedge inSecurityPerimeter(p) \wedge type(o, wifi) \rightarrow permission(s, use, o)$
- The second rule shows that ABAC allows expressing rules where rights can be granted to users without needing them to authenticate
  - Security condition evaluates to True if the attributes have been authenticated

Access Controls Models / ABAC

- ABAC rules are written with a policy language
- XACML is an XML-based language standard for specifying ABAC rules
- XACML offers several strategies for solving conflicts between permissions and prohibitions
  - Deny overrides, first applicable etc.
- The XACML standard defines also a processing model for evaluating access requests according to ABAC rules

Access Controls Models / ABAC



By Axiomatics - Axiomatics, CC BY 3.0, https://commons.wikimedia.org/w/index.php?curid=48397652

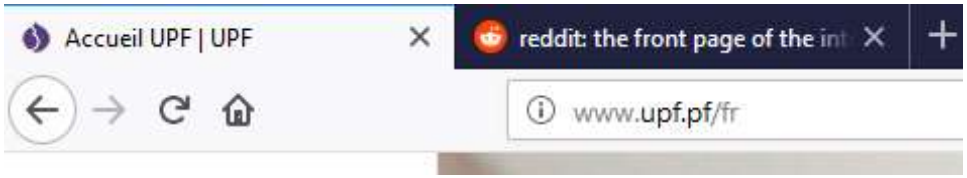
Resource sharing using a Cross-Origin Resource Sharing (CORS) policy

DATUM ACADEMY

- Web browser Security
- CORS

CORS / Web Browser Security

- Web browser security deals with the security of the client side of a web application
- A browser may run several Web applications
- Basically, the purpose of web browser security is to,
  - Protect the computer from potentially malicious code included in downloaded html pages
  - Isolate the content of these different applications from each other



CORS / Web Browser Security

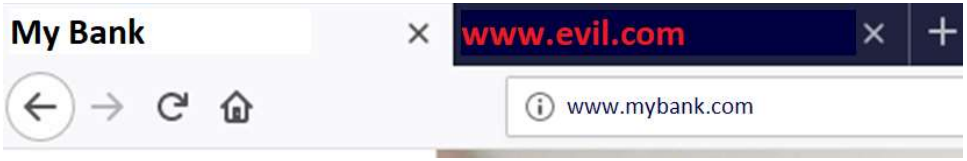
- To protect the computer from malicious code included in downloaded html pages, downloaded scripts run in a sandbox
- A browser sandbox severely restricts script access to host computer resources
  - E.g. Downloaded scripts cannot create or write files
  - Reading a file is only allowed by letting the user select the file

CORS / Web Browser Security

- To isolate the content of the different browser tabs from each others, downloaded scripts are constrained by the same-origin policy
- The same-origin policy says that a document or script loaded from one origin cannot interact with a resource from another origin.
- Most modern browser enforce the same-origin policy

CORS / Web Browser Security

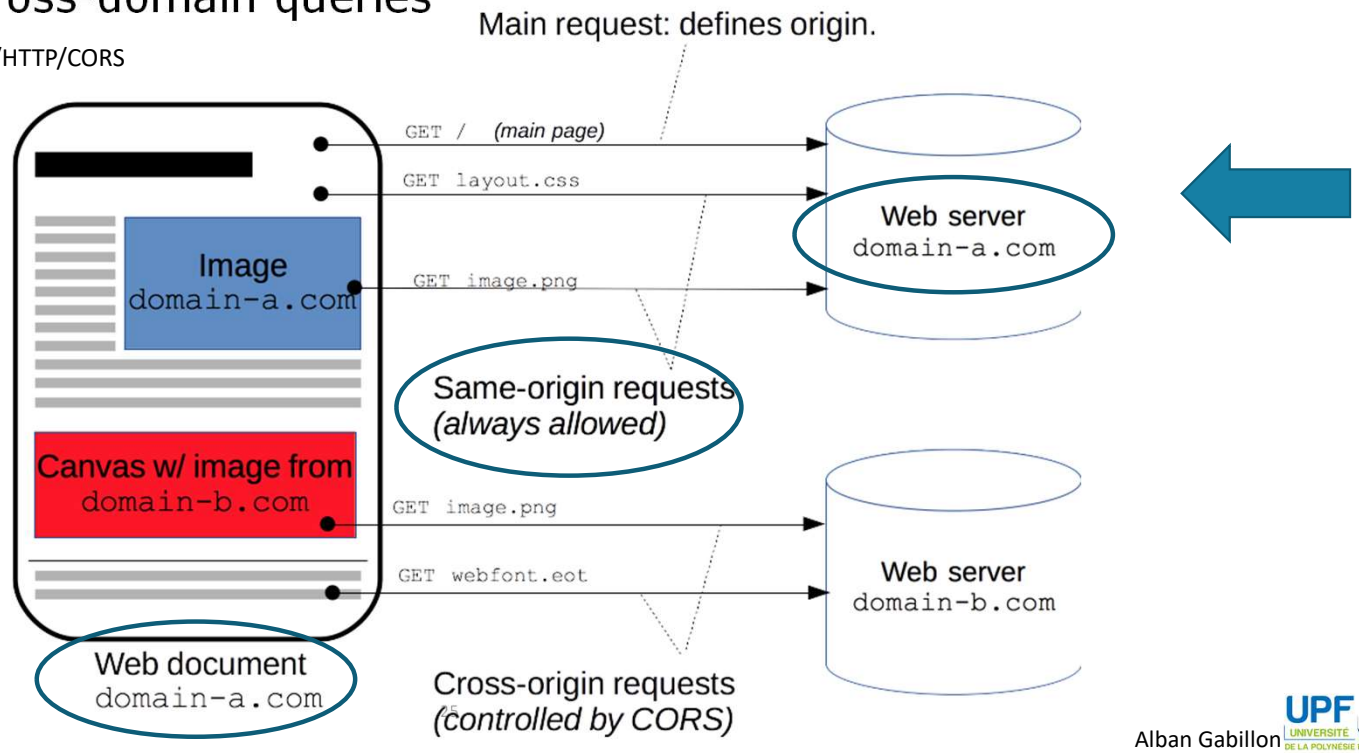
- Without the same origin policy, a malicious script loaded from www.evil.com could use the authentication cookie related to www.mybank.com and order some fraudulent transactions on my behalf



CORS

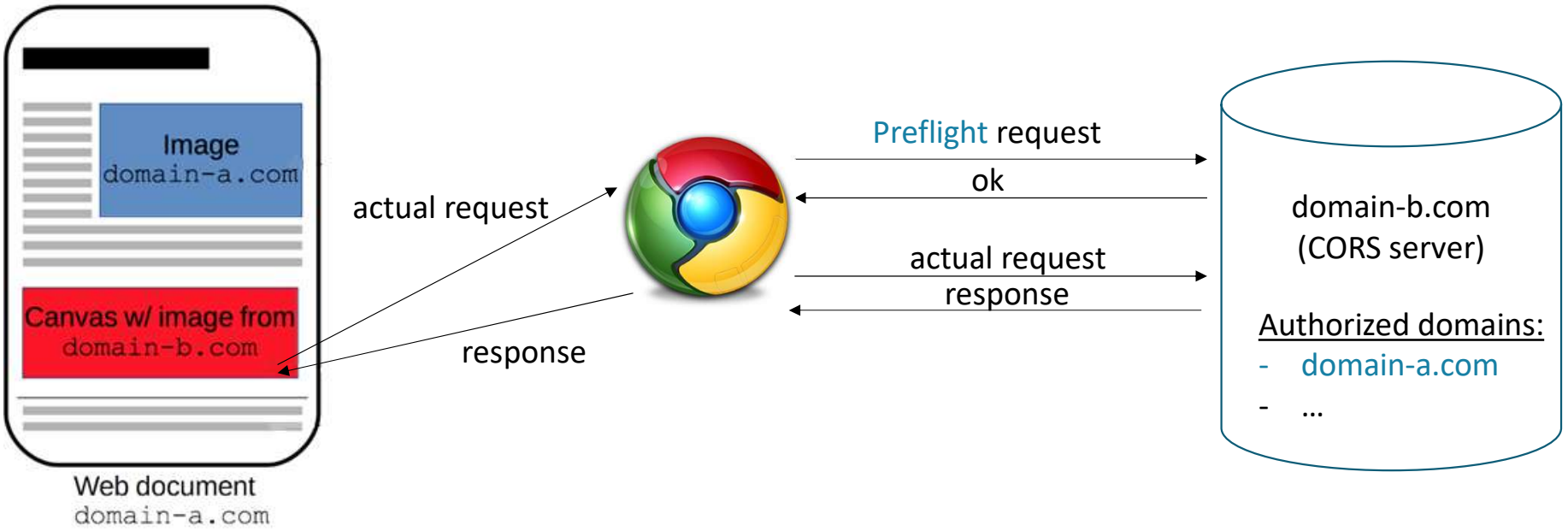
- However, when creating complex client-side applications, it is usually necessary to perform Ajax cross-domain queries

https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS



CORS

- The `domain-b` server must be a CORS server and it must allow `domain-a` to access to its resources

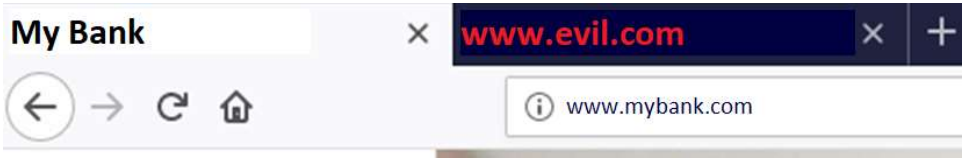


CORS

- The response to the preflight request sent by the CORS server includes a list of specific CORS http headers
  - The value of these headers indicate whether the CORS server will accept the actual request
- If the response to the preflight request is positive the browser then sends the actual request
  - Otherwise, the actual request is blocked by the browser
- A CORS server maintains a whitelist of authorized domains unless it exposes a public API
  - in that case it accepts requests sent by clients from any domain

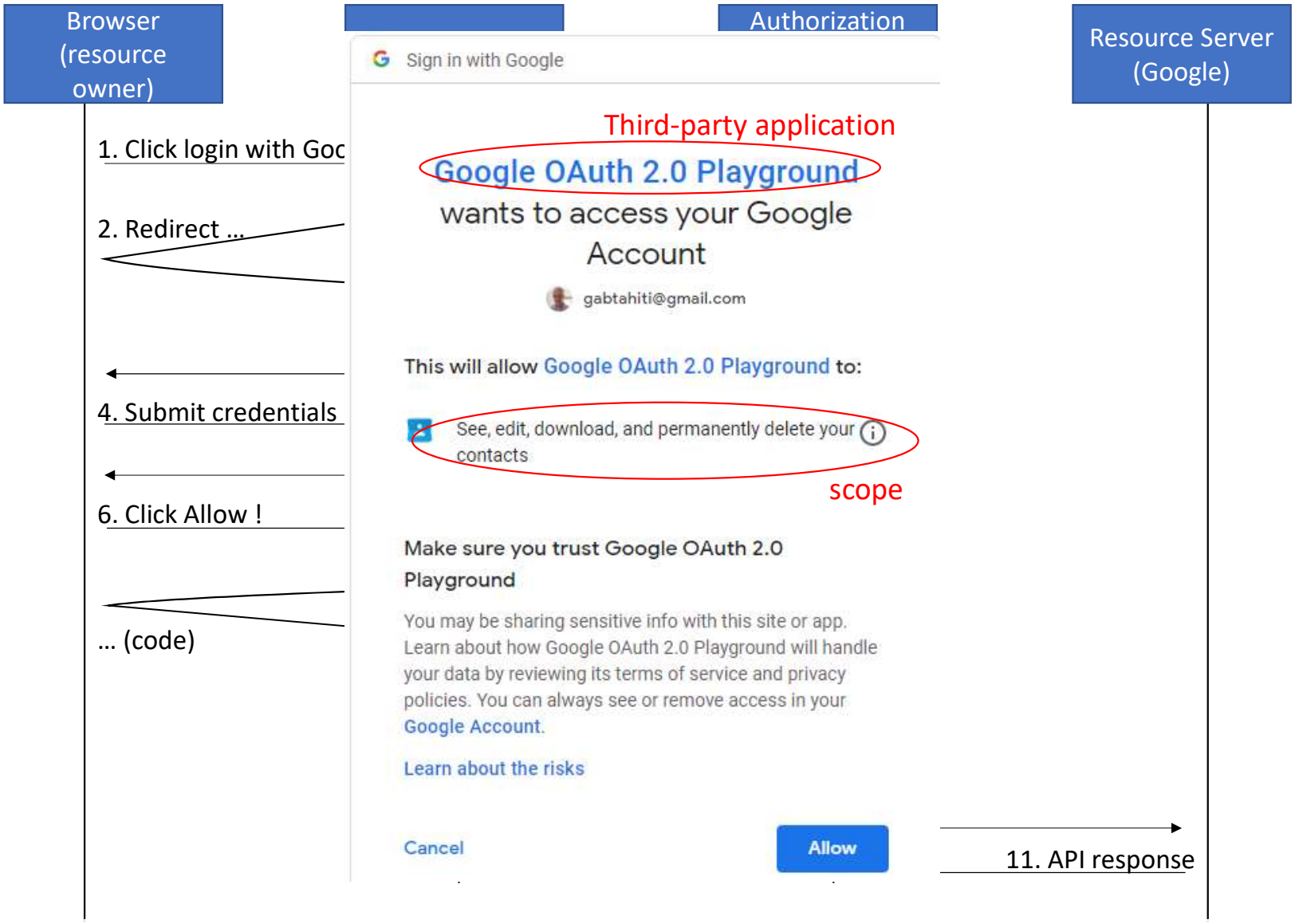
CORS

- The CORS policy is a DAC policy addressing domains whose default policy is the mandatory same-origin policy
- The CORS policy is defined by the CORS server (owner of the resources) and enforced by the browser (trusted component)
- Of course, if `mybank.com` is a CORS server then it does not have `evil.com` in its whitelist of authorized domains
  - Same-origin policy applies



Access delegation using OAuth 2.0

- Suppose an application wants to access your Google profile
- A simple but totally insecure solution would be to provide the application with your Google credentials
- OAuth 2.0 is another solution commonly used on Internet grant access to resources without disclosing credentials
- OAuth is used by companies such as Google, Facebook, or Twitter to allow users to securely share information about their accounts with third-party applications


Browser (resource owner)
Authorization
Resource Server (Google)
1. Click login with Goc
2. Redirect …
4. Submit credentials
6. Click Allow !
… (code)
Sign in with Google
Third-party application
Google OAuth 2.0 Playground
wants to access your Google Account
gabtahiti@gmail.com
This will allow Google OAuth 2.0 Playground to:
See, edit, download, and permanently delete your contacts
scope
Make sure you trust Google OAuth 2.0 Playground
You may be sharing sensitive info with this site or app. Learn about how Google OAuth 2.0 Playground will handle your data by reviewing its terms of service and privacy policies. You can always see or remove access in your Google Account.
Learn about the risks
Cancel
Allow
11. API response

Access delegation using OAuth 2.0

- No access token is sent back to the browser which is not considered totally secure. The access token is sent to the upf.pf application server instead.
- Depending on the case, the access token can be,
  - short term (valid only for the current session)
  - long term (valid for several days) and can therefore be used by the third-party application when the user is offline.
- The authorization server does not have to be a CORS server since it communicates directly with the browser and not with any JavaScript. The same applies to the resource server since it only interacts with the application server.

Access delegation using OAuth 2.0

- The access control model underlying the OAuth protocol is the DAC model in which users grant permissions to third parties to access their personal data.
- A permission is implemented as an access token i.e., a capacity.
- It is possible to combine OAuth with XACML to handle ABAC dynamic authorizations although there is no standard way of doing it.
- OAuth ➔ Authorization delegation
- OAuth + OpenId Connect ➔ Authentication delegation
  - Allow Web Single Sign-On (SSO)

Access delegation using OAuth 2.0

- The OAuth protocol can be used to build APIs that comply with the General Data Protection Regulation (GDPR).
- Indeed, the GDPR imposes on processing entities the obligation to obtain the consent of the subject entities before using their personal data.
- OAuth must, however, be implemented with the token revocation extension (RFC 7009) to allow users to revoke an access token at any time since the withdrawal of consent is also one of the subject's rights under the GDPR.
- Practice OAuth at https://developers.google.com/oauthplayground/

Identity and Access Management (IAM) policies in the cloud

DATUM ACADEMY

- Cloud providers like Amazon AWS, Google Cloud Platform or Microsoft Azure offer cloud ecosystems i.e., complex systems of interdependent assets
- A single cloud account (IaaS) can be used to set up a virtual organization managing several users and resources like databases, virtual machines, virtual networks etc.
- The main security component of a cloud ecosystem is the Identity and Access Management (IAM).
- The IAM is a web service to manage authentications and authorization policies in the virtual organization.

Identity and Access Management (IAM) policies in the cloud

DATUM ACADEMY

- Access control model that approximately represents the security models implemented in the various existing cloud platforms
- This access control models combines features from several access control paradigms
  - Subjects (user or application)
  - Resources (basically anything that needs to be protected)
  - Authorization policies (permissions or prohibitions)
  - User groups (set of subjects)
  - Roles (set of authorization policies)

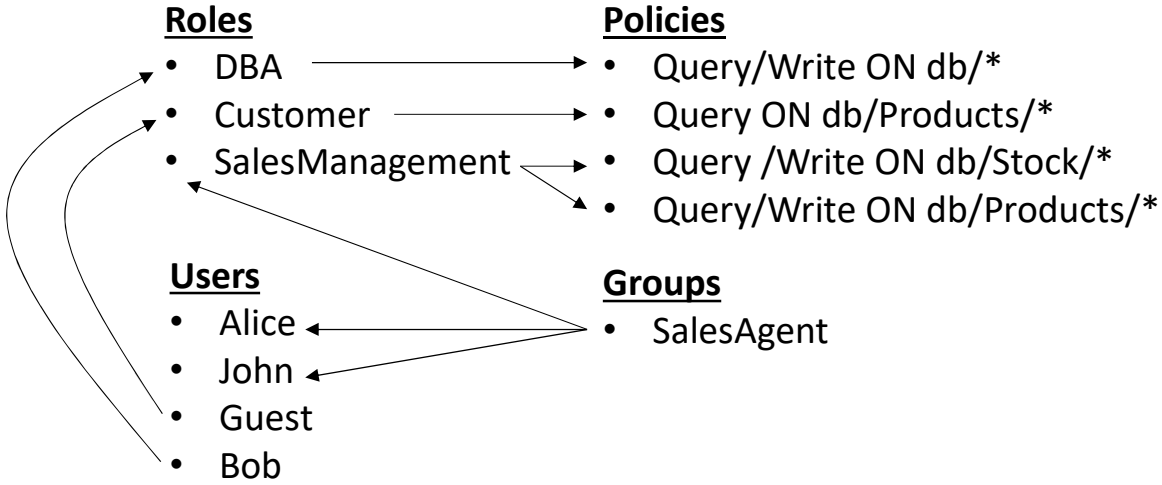
Identity and Access Management (IAM) policies in the cloud

DATUM ACADEMY

- Subjects can belong to more than one group
- Resources are hierarchically organized
- A security policy is the authorization or the prohibition to do a set of actions on a set of resources.
  - It possibly includes a contextual condition.
- A policy can be attached to a subject, a group, or a role
- Roles can be hierarchically organized.
- Roles are assigned to subjects or groups.

Identity and Access Management (IAM) policies in the cloud

- The conflict resolution policy is quite simple and can be summarized as follows:
  - in case of conflict between two policies, denial prevails.
- The default policy is denial
- Small Case Study



Identity and Access Management (IAM) policies in the cloud

DATUM ACADEMY

- IAM security model is essentially an RBAC security model with features borrowed from other access control paradigms
- A set of users, a set of low-level resources and a set of policies can be represented as a group, a high-level resource, and a role, respectively
  - This clustering of entities is a characteristic of the OrBAC model
- Processing model?

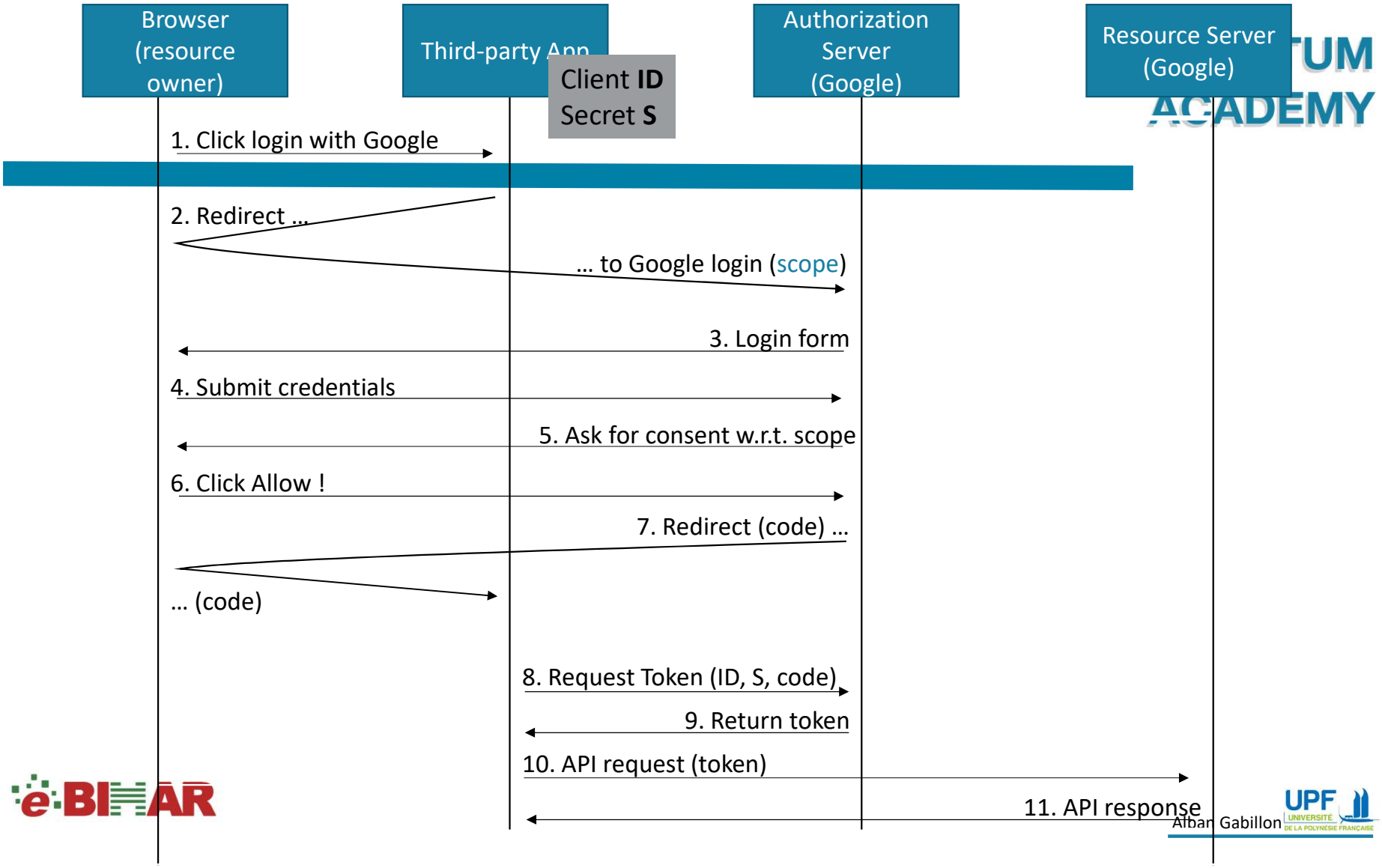
Identity and Access Management (IAM) policies in the cloud

- Regarding cloud security, there is one big issue that need to be remedied.
- How can the cloud customer trust the cloud provider?
  - Homomorphic encryption is a solution that would allow encrypted data to be processed. However, it is challenging to implement.
  - IBM fully Homomorphic Encryption Toolkit :
    - https://github.com/IBM/fhe-toolkit-linux
  - Homomorphic encryption does not solve all the problems since the code itself can convey sensitive information and should also be protected from the cloud provider.

Web Access Control Strategies

DATUM ACADEMY

- Thank you


Browser (resource owner)
Third-party App
Client ID
Secret S
Authorization Server (Google)
Resource Server (Google)
1. Click login with Google
2. Redirect …
… to Google login (scope)
3. Login form
4. Submit credentials
5. Ask for consent w.r.t. scope
6. Click Allow !
7. Redirect (code) …
… (code)
8. Request Token (ID, S, code)
9. Return token
10. API request (token)
11. API response

Access Controls Models / MAC

- The Multilevel security policy aims at preserving the confidentiality of data
  - Each user receives a confidentiality (sensitivity) level called a clearance level
    - This level measures the degree of trust one can have in the user
  - Each object receives a confidentiality level called a classification level
- Clearance levels and classification levels are derived from a single set of totally ordered sensitivity levels
  - $Unclassified < Confidential < Secret < Top\ Secret$
  - $l_1 < l_2$ reads « l_2 strictly dominates l_1»

Multilevel security

- The Multilevel Security policy is mandatory and can easily be expressed as follows:
  - Let us define an entity as a subject or an object
  - Let $level(e)$ be the sensitivity level of entity e
  - Let e_1 and e_2 be two entities

Information flow from e_1 to e_2 is permitted if $level(e_1) \leq level(e_2)$
Information flow from e_1 to e_2 is prohibited if $level(e_1) > level(e_2)$

Multilevel security

- However, regarding read and write operations, Bell & LaPadula (1973) have shown that the two following properties are necessary (but not sufficient) to enforce the multilevel security policy:
  - No read-up: A low-level process cannot read from a high-level file
  - No write-down: A high-level process cannot write to a low-level file
  - Implicitly, other read/write operations (read-same, read-down, write-same, write-up) are permitted