Access control unveiled: Challenges & best practices

ELIMITY

SecAppDev

Maarten Decat
Co-founder & CEO
maarten@elimity.com
www.elimity.com

The topics of this presentation

Access control

&

Identity & Access Management

Outline

1. **Introduction**
   a. **What is access control?**
   b. **What is IAM?**
2. Deeper dive into access control
3. Deeper dive into IAM
4. How to IAM and access control relate?
5. Conclusion

What is access control?

Access control is the part of a system that constrains the *actions* that are performed in a system based on *access control rules*.

- As with any security: confidentiality, integrity, availability
- Layer in between (malicious) users and the protected system
- Part of the Trusted Computing Base

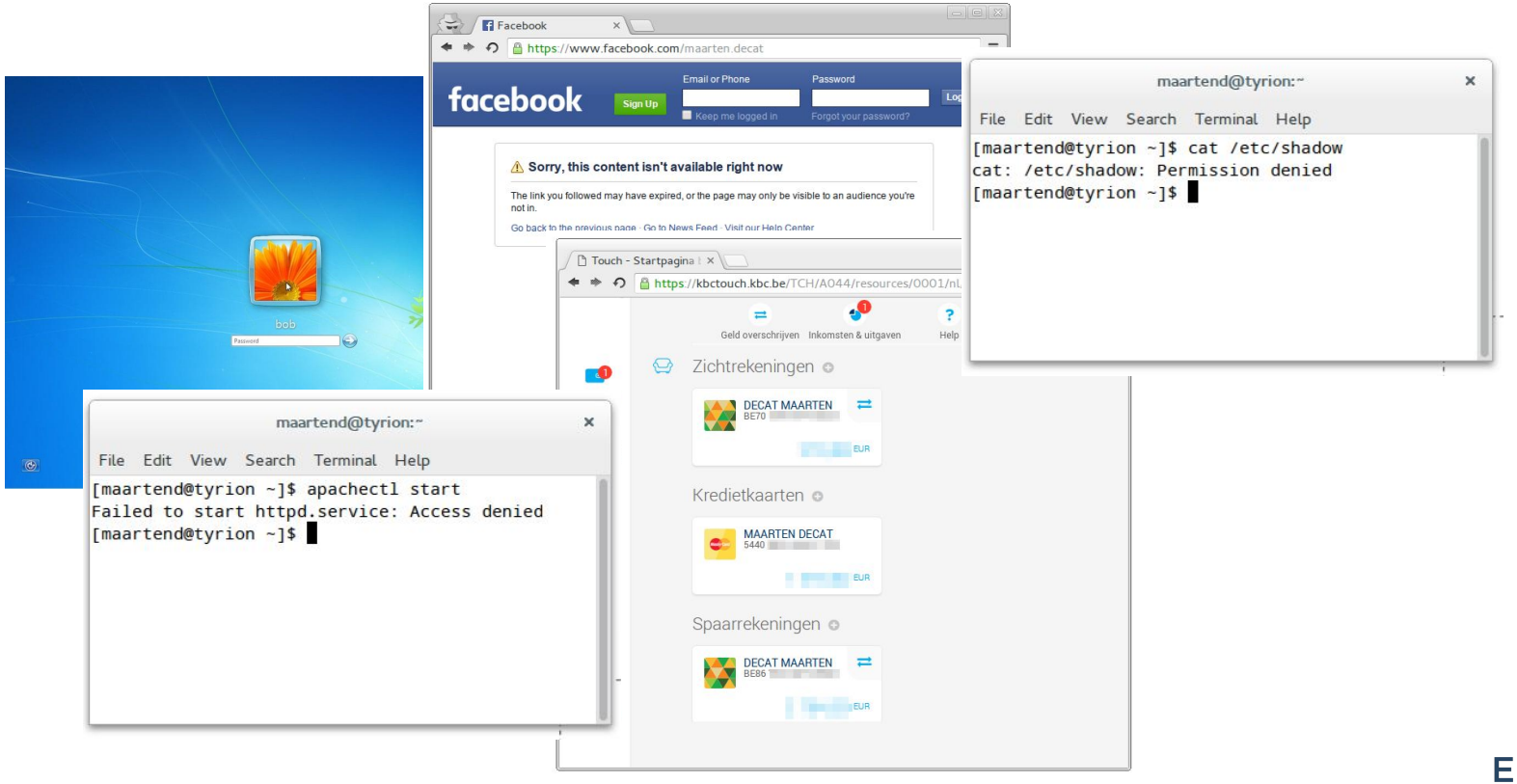
What is access control?

1. Not easy to get right,
   e.g., what about windows?
2. Difference between access rules and mechanism
3. Different mechanisms have different properties
4. Different mechanisms support different rules

Access control in the physical world

ELIMITY

Access control in software



ELIMITY

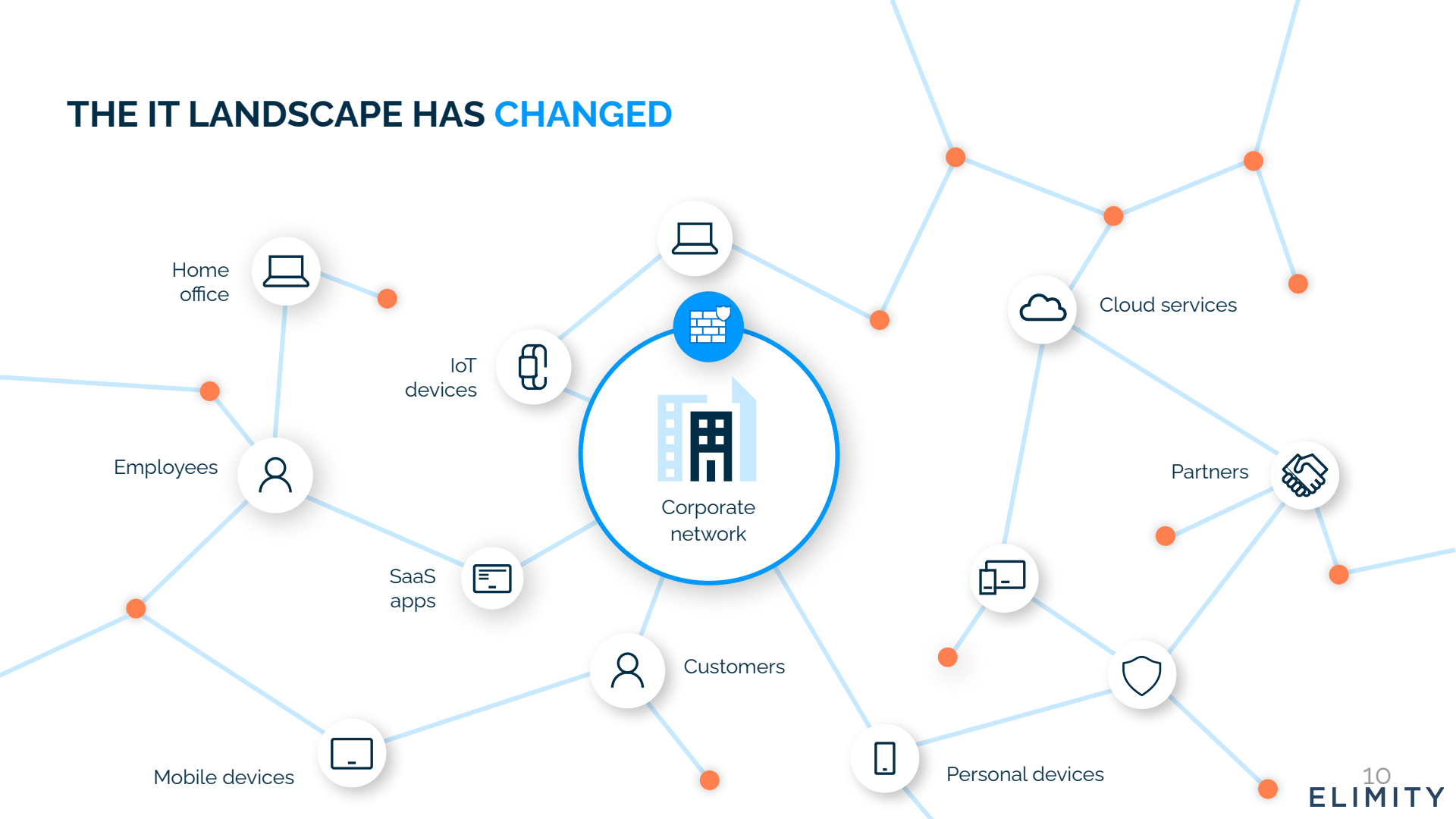
What is IAM?

Identity & Access Management (IAM)
encompasses all processes used by an organization
to ensure that everyone can access the data they need
and only the data that they need.

What is IAM?

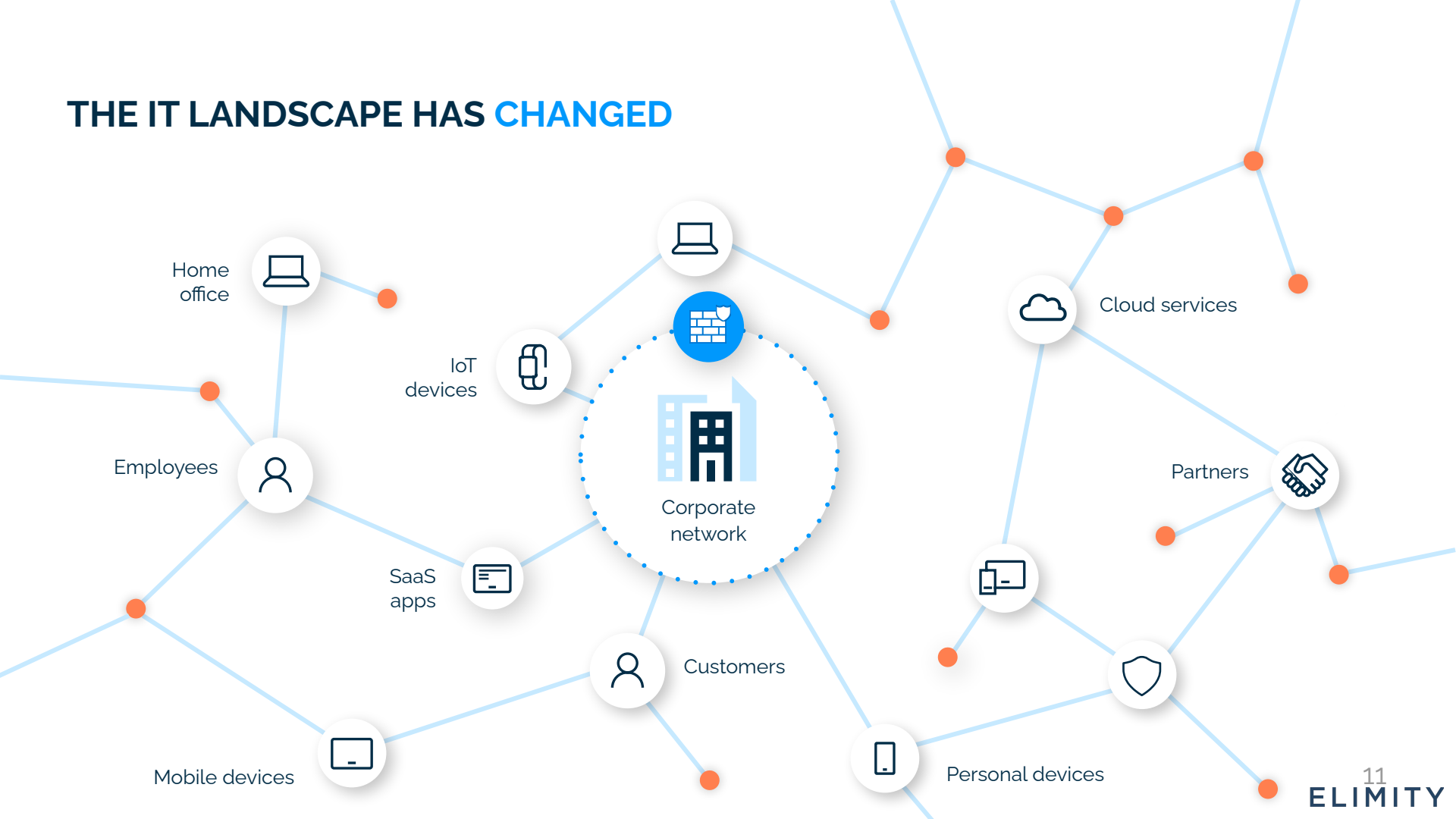
ELIMITY

THE IT LANDSCAPE HAS CHANGED



ELIMITY

THE IT LANDSCAPE HAS CHANGED



ELIMITY

THE IT LANDSCAPE HAS CHANGED

WHAT STILL CONNECTS ALL THE PIECES:

IDENTITY

94%

of organizations have had an identity-related security breach

TAKING CONTROL OVER

WHO CAN ACCESS WHICH DATA & APPLICATIONS

IS ESSENTIAL FOR CYBER SECURITY

ISO 27001	A.9 Access Control
NIST 800-53	Control family: Access Control
CIS CONTROLS	14. Controlled Access Based on the Need to Know

ISO27001 | NIST | SOC2 | NIS | SOX | GDPR | CIS

ELIMITY

Side note: Security is not the only driver for IAM

1. CYBERSECURITY

Minimize the chances of credential theft

Minimize the impact of credential theft (enforce least privilege)

2. COMPLIANCE

Comply to standards and regulations

Avoid unneeded access (e.g., GDPR)

Show that you are in control (audit trails)

3. OPERATIONAL EFFICIENCY

Improve time-to-work

Decrease burden on helpdesk

Automate provisioning

Automate password resets

Access control & IAM

Even though access control is important, it is #1 on OWASP Top 10.

Even though almost every hack starts with stolen credentials, many organizations are still not in control of their users and accesses.

The rest of this presentation: go deeper into access control and go deeper into IAM to give you the tools to better protect the data in your application and help your customers protect their data in your application.

Outline

1. Introduction
2. **Deeper dive into access control**
   a. What is access control?
   b. Challenges
   c. Access control models
   d. How to implement
3. Deeper dive into IAM
4. How to IAM and access control relate?
5. Conclusion

Outline

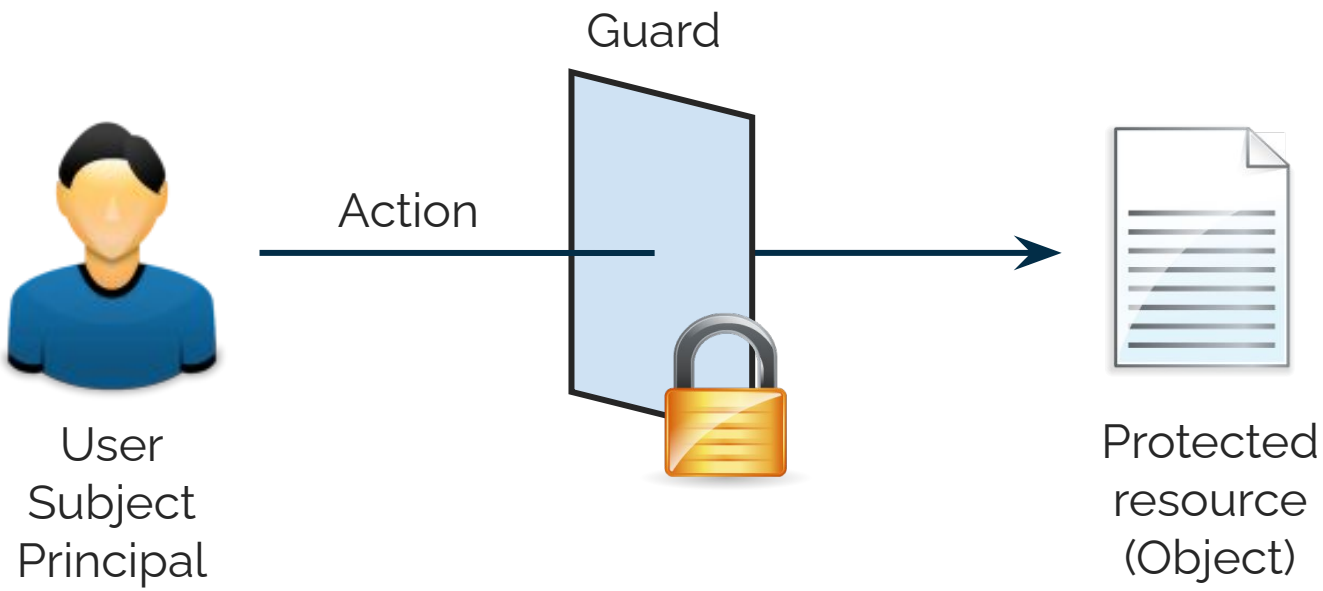
1. Introduction
2. **Deeper dive into access control**
   a. **What is access control?**
   b. Challenges
   c. Access control models
   d. How to implement
3. Deeper dive into IAM
4. How to IAM and access control relate?
5. Conclusion

What is access control?

Access control is the part of a system that constrains the *actions* that are performed in a system based on *access control rules*.

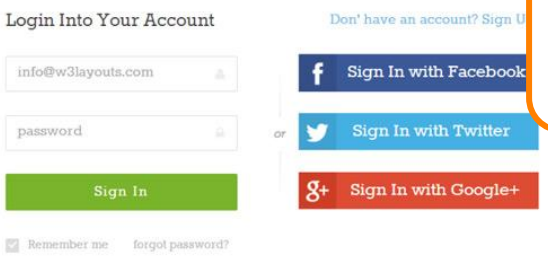
- As with any security: confidentiality, integrity, availability
- Layer in between (malicious) users and the protected system
- Part of the Trusted Computing Base

10,000m point of view



ELIMITY

But there is more to it



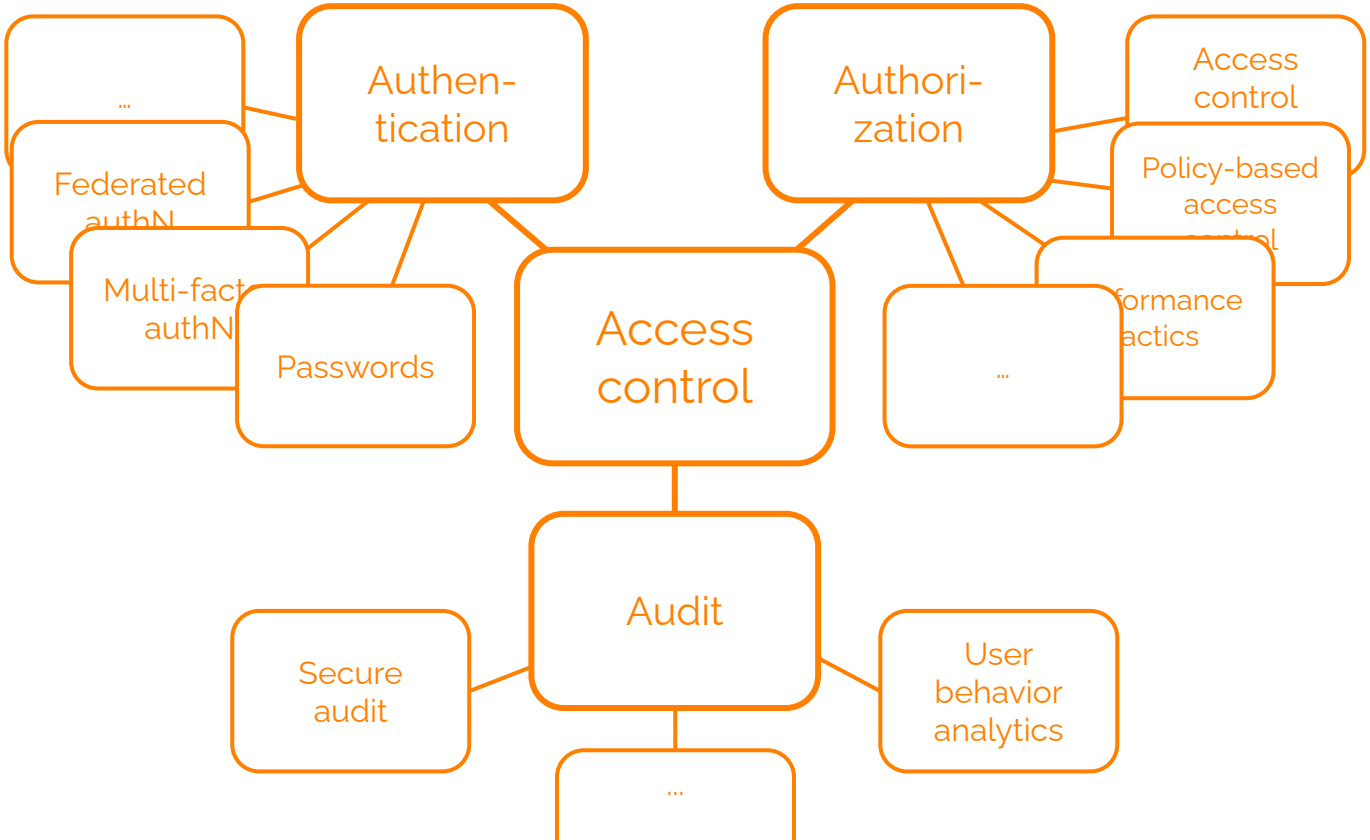
Authen-
tication

Authori-
zation

Access
control

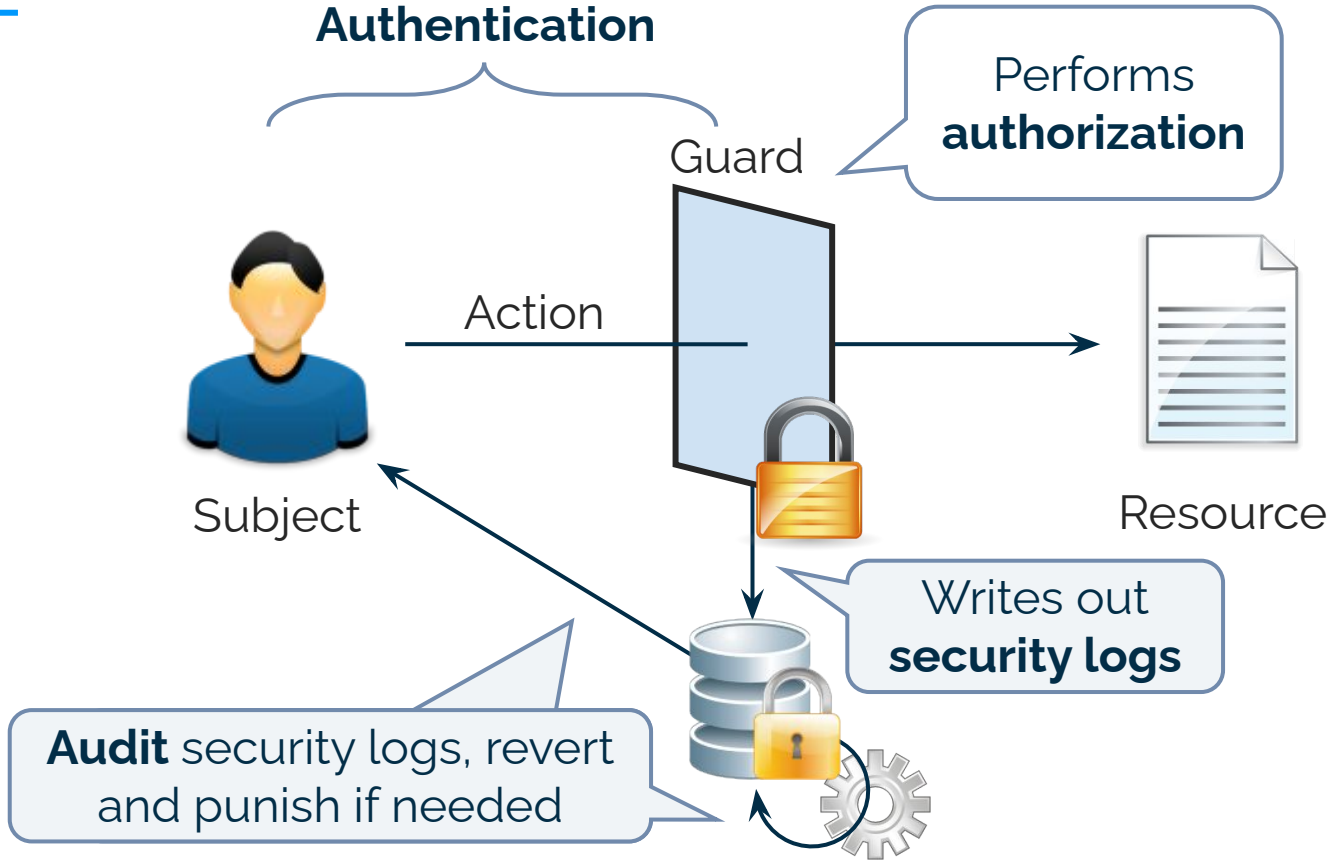
Audit

But there is more to it



ELIMITY

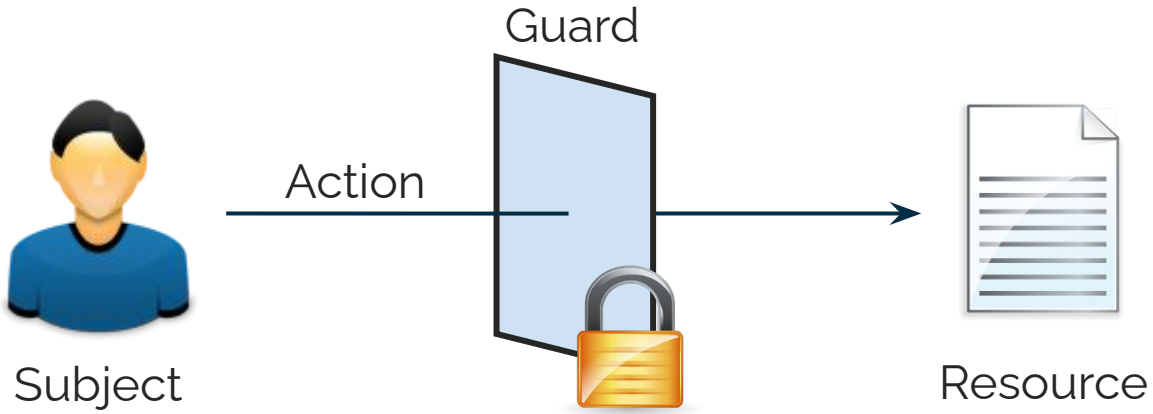
5000m point of view



ELIMITY

For the rest of this presentation

“Access control” = “authorization”



ELIMITY

Outline

1. Introduction
2. **Deeper dive into access control**
   a. What is access control?
   b. Challenges
   c. Access control models
   d. How to implement
3. Deeper dive into IAM
4. How to IAM and access control relate?
5. Conclusion

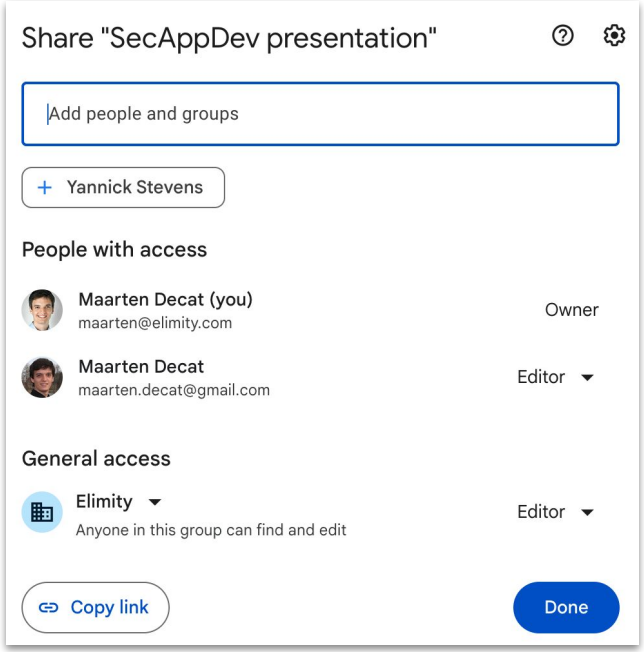
Models, policies and mechanisms

- **Guard** is responsible for mediating access
  - Authorize specific actions
  - *Mechanism* that enforces specific *security rules*
- Rules, policies, models and mechanisms
  - **Access rules**: the logical access rules, independent of representation
  - **Mechanism**: low-level implementation of controls
  - **Model**: (formal) representation of how rules can be expressed
- Access control seems straightforward… but is it?

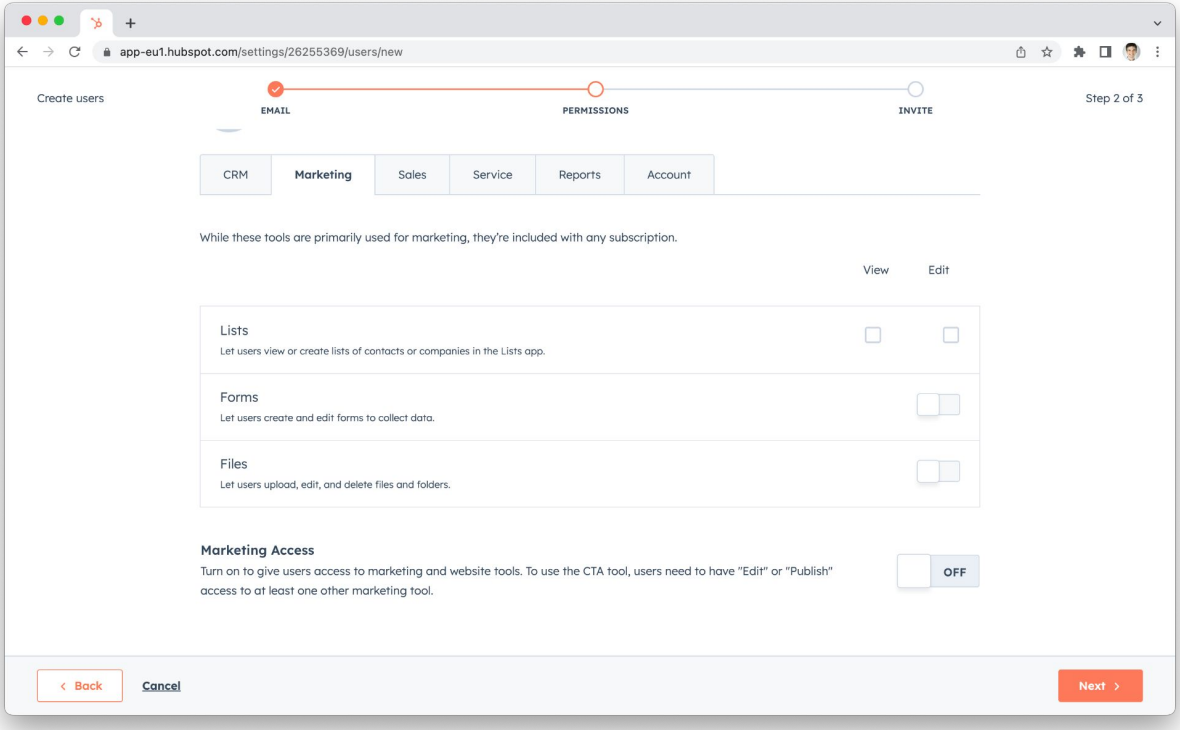
Example access control model:

A user is permitted to read/comment/write a file if any of the following holds:

1. he/she is the owner of the file,
2. he/she has explicitly been given this permission or higher,
3. he/she is part of a mail group that has explicit been given this permission or higher,
4. the file has been shared with the whole organization of the user that created it, and the user is part of that organization and the default permission for the organization is this permission or higher,
5. the file has been link-shared for that permission and the he/she has opened the file using that link.

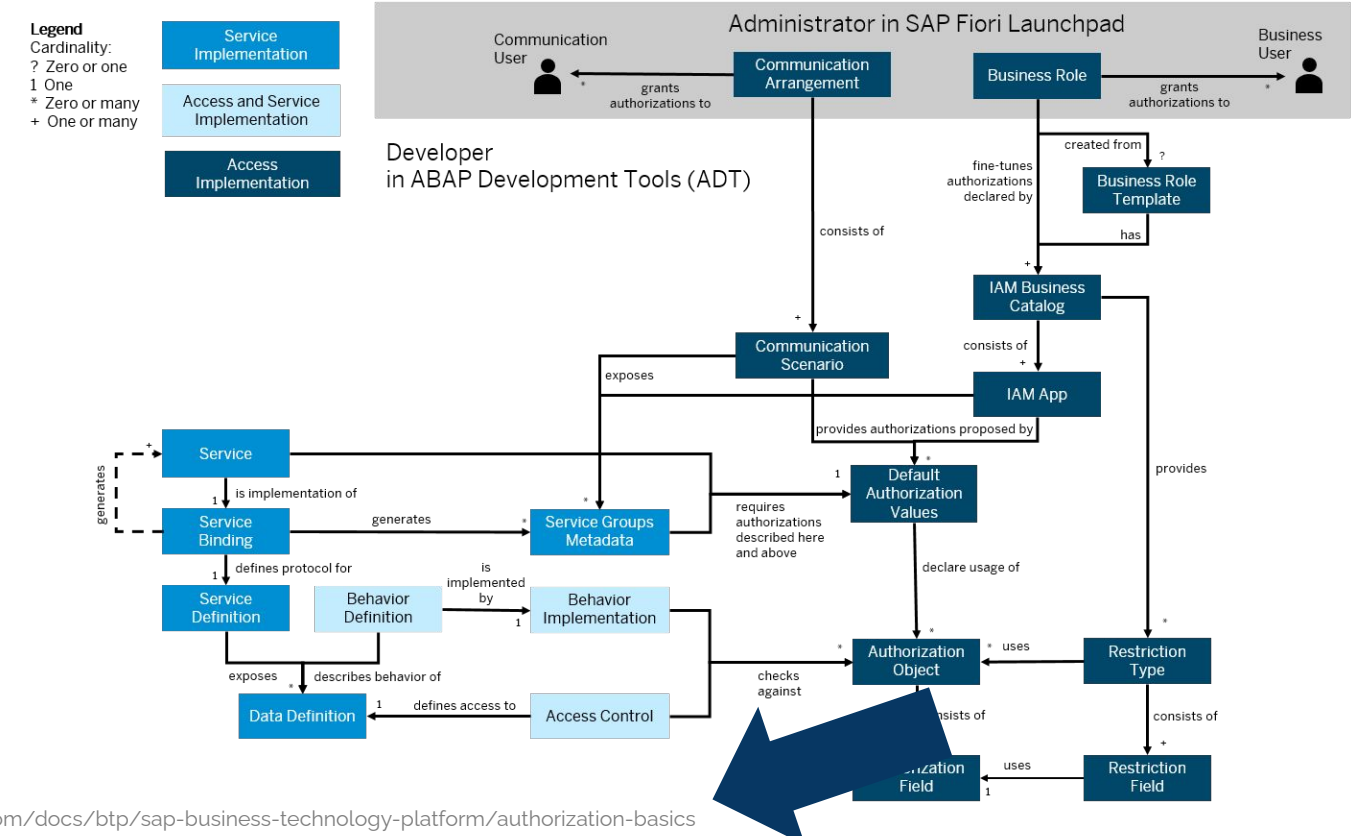


Examples access control model: HubSpot



ELIMITY

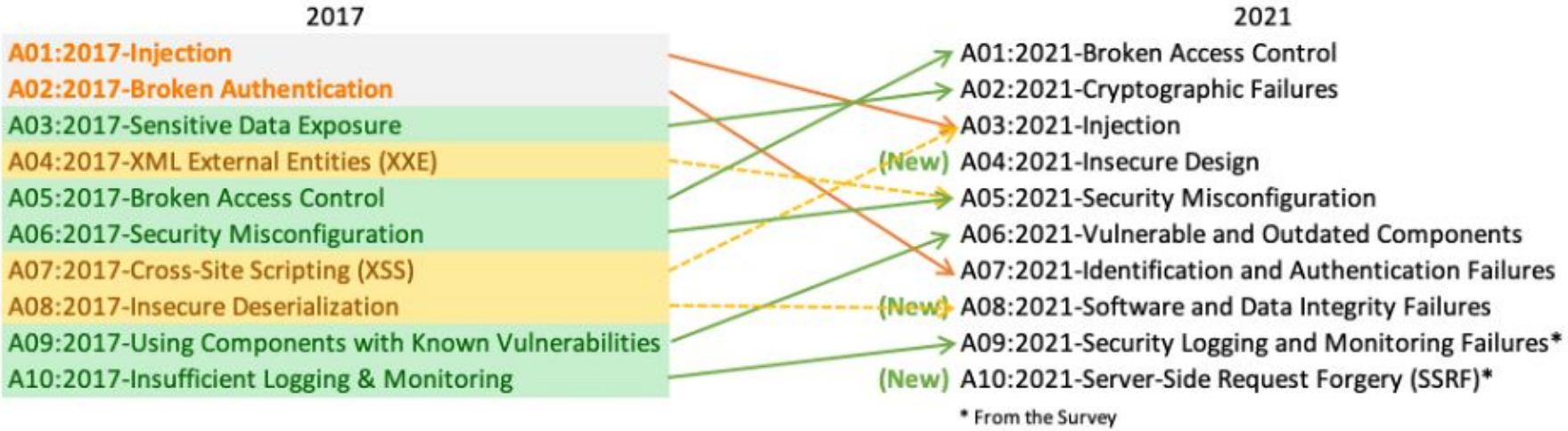
Examples access control model:



Source: https://help.sap.com/docs/btp/sap-business-technology-platform/authorization-basics

ELIMITY

The result: #1 in OWASP Top 10



"A01:2021-Broken Access Control moves up from the fifth position to the category with the most serious web application security risk; the contributed data indicates that on average, 3.81% of applications tested had one or more Common Weakness Enumerations (CWEs) with more than 318k occurrences of CWEs in this risk category. The 34 CWEs mapped to Broken Access Control had more occurrences in applications than any other category." Source: https://owasp.org/Top10/

OWASP Top 10: A01 Broken Access Control

Common access control vulnerabilities include:

1. **Violation of the principle of least privilege** or deny by default, where access should only be granted for particular capabilities, roles, or users, but is available to anyone.
2. **Bypassing access control checks** by modifying the URL (parameter tampering or force browsing), internal application state, or the HTML page, or by using an attack tool modifying API requests.
3. **Permitting viewing or editing someone else's account**, by providing its unique identifier (insecure direct object references)
4. Accessing **API with missing access controls** for POST, PUT and DELETE.
5. **Elevation of privilege**. Acting as a user without being logged in or acting as an admin when logged in as a user.
6. **Metadata manipulation**, such as replaying or tampering with a JSON Web Token (JWT) access control token, or a cookie or hidden field manipulated to elevate privileges or abusing JWT invalidation.

OWASP Top 10: A01 Broken Access Control

How to prevent:

1. Only **rely on trusted server-side code** or server-less API, where the attacker cannot modify the access control check or metadata.
2. Except for public resources, **deny by default**.
3. Implement access control mechanisms once and **re-use them throughout the application**.
4. Model access controls should enforce record **ownership** rather than accepting that the user can create, read, update, or delete any record.
5. Unique application business limit requirements should be enforced by domain models.
6. **Rate limit API** and controller access to minimize the harm from automated attack tooling.
7. Stateful **session identifiers should be invalidated** on the server after logout.

=> No silver bullets: apply decent engineering, high-quality testing, KISS

Outline

1. Introduction
2. **Deeper dive into access control**
   a. What is access control?
   b. Challenges
   c. **Access control models**
   d. How to implement
3. Deeper dive into IAM
4. How to IAM and access control relate?
5. Conclusion

The basics: the access control matrix

Resources

Subjects

	File A	File B	File C
Alice	read	read	read
Bob	read, write		
Charlie		read, write	read, write

Permissions

[Lampson1971]

ELIMITY

Extensions of the access control matrix:

Who can assign permissions?

Who can assign permissions?

In general, two approaches:

1. Mandatory access control (MAC)
   - By central authority
2. Discretionary access control (DAC)
   - By subjects themselves

Mandatory access control (MAC)

- Permissions are assigned by a central authority according to a central policy
  - Good fit within organizations and systems with a strong need for central controls
  - Low flexibility and high management overhead
- Mandatory Access Control in use
  - Often linked to multi-level security systems -> see later on
    - E.g. Government-regulated secrecy systems, military applications
  - Modern operating systems, to separate applications and processes
    - E.g. Windows' *Mandatory Integrity Control*, SELinux, TrustedBSD
  - The essence of every IAM security strategy

Discretionary access control (DAC)

- Permissions are set *at the discretion* of the subjects, e.g., the resource owner
  - Highly flexible policy, where permissions can be transferred
  - Lack of central control makes revocation or changes difficult

- Discretionary access control in use
  - Controlling access to files
    - E.g., Windows Access Control Lists (ACL), UNIX file handles, Teams, Google Drive, ...
  - Controlling the sharing of personal information
    - E.g., Social networks

Recap: MAC vs DAC

- Two dual approaches
- In practice: combine both
  - Provide some form of discretionary self-management within the constraints of mandatory access rules
    - For example, delegate administration of team resources to an administrator
  - Options:
    - Trust subjects to enforce mandatory policy
    - Audit mandatory policy
    - Enforce mandatory policy
- My experience:
  - DAC in an enterprise context gives many CISOs a headache. Just think of the file shares at Stad Antwerpen

Extensions of the access control matrix:

How are permissions assigned?

Existing models

- Identity-based access control
- Multi-level access control
- Role-based access control (RBAC)
- Attribute-based access control (ABAC)

Identity-based access control

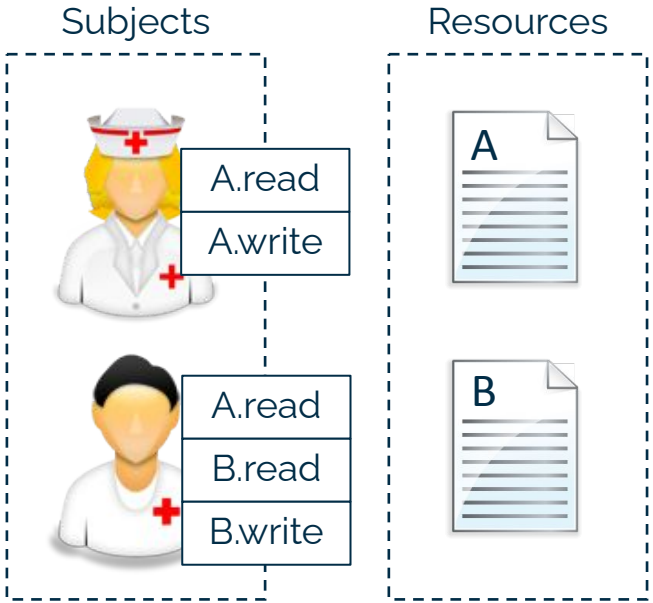
- Assign permissions to individual subjects and resources
  - This is actually again the Access Control Matrix

	File A	File B	File C
Alice	read	read	read
Bob	read, write		
Charlie		read, write	read, write

Identity-based access control

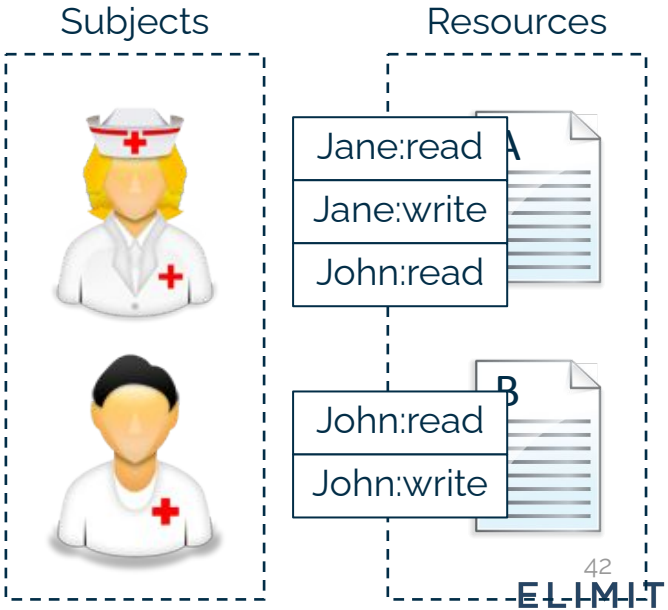
Possible implementations: store 1 big matrix (not efficient) or:

Access Control Lists



	File A	File B
Jane	Read Write	
John	Read	Read Write

Capability Lists



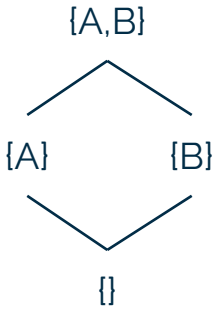
ELIMITY

Identity-based access control

- Advantage: flexibility
- Disadvantage: Large management effort
  - E.g., “all nurses can read patient files” -> repeat for all nurses
  - E.g., “patients can read their own patient files” -> repeat for all patients
- Used in practice
  - E.g., Google Drive

Multi-level access control

- Sometimes also called Lattice-Based Access Control
- Strict control over information flow
  - Resources are assigned security classifications
  - Subjects (and their programs) are assigned security clearances
  - These labels are organized in a lattice
- Two well-known rule sets:
  - Bell-LaPadula (confidentiality)
  - Biba (integrity)

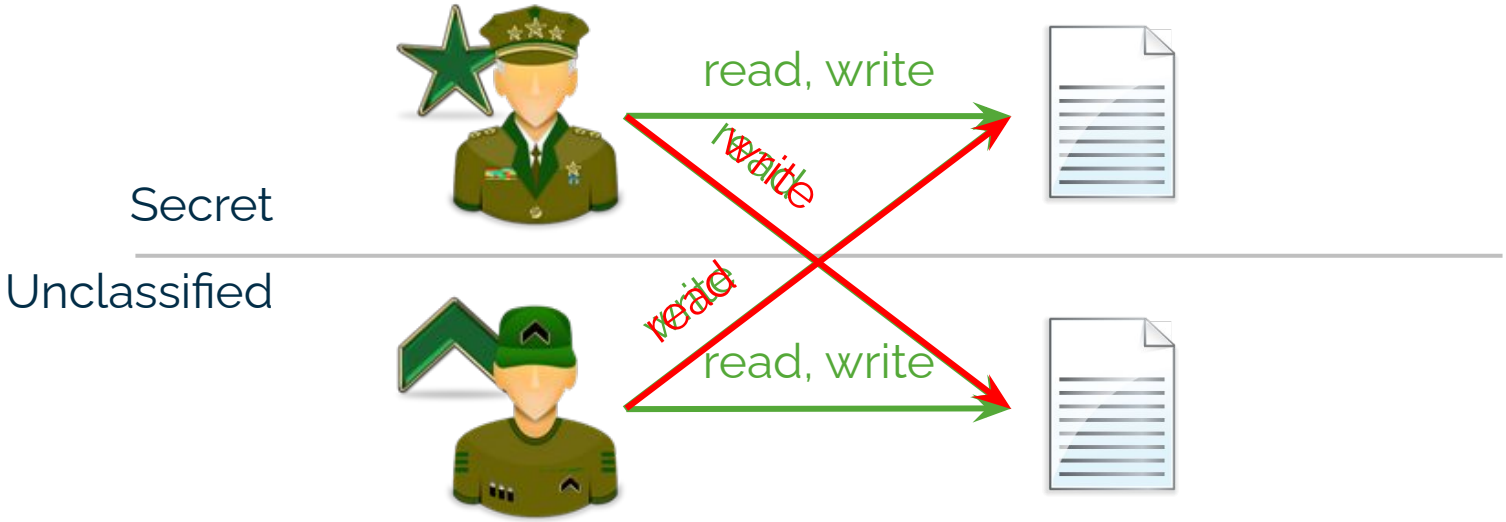


ELIMITY

Multi-level access control

- Model of Bell-LaPadula:
  - No read up
  - No write down ("✰-property")

Confidentiality

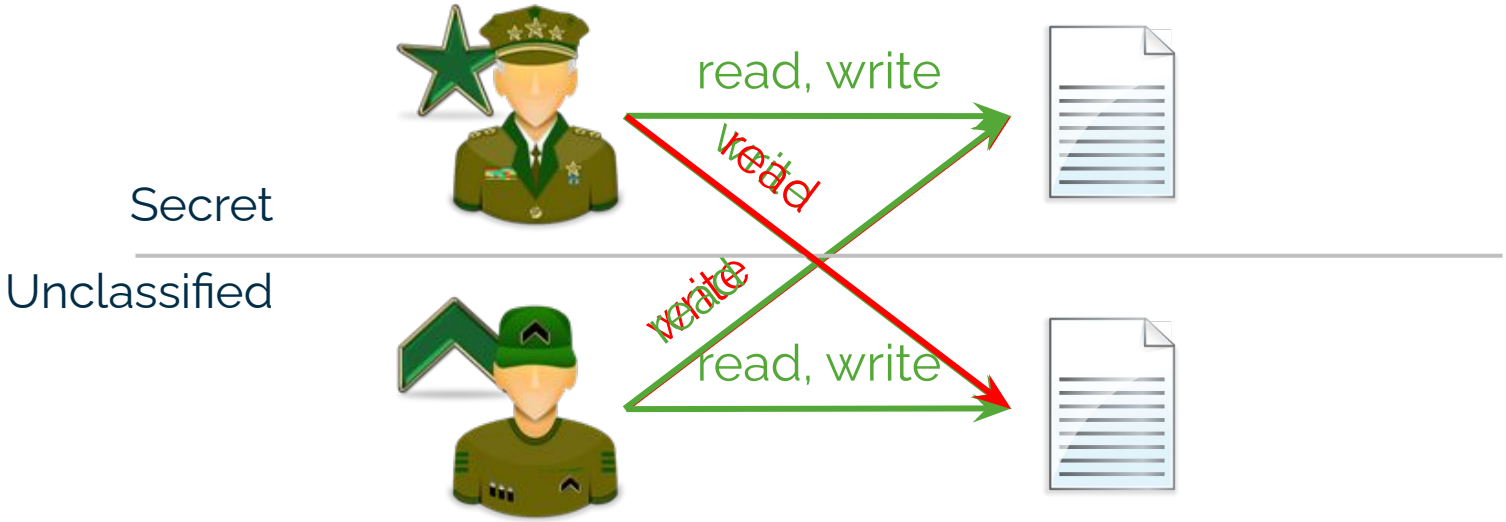


ELIMITY

Multi-level access control

- Model of Biba:
  - No write up
  - No read down

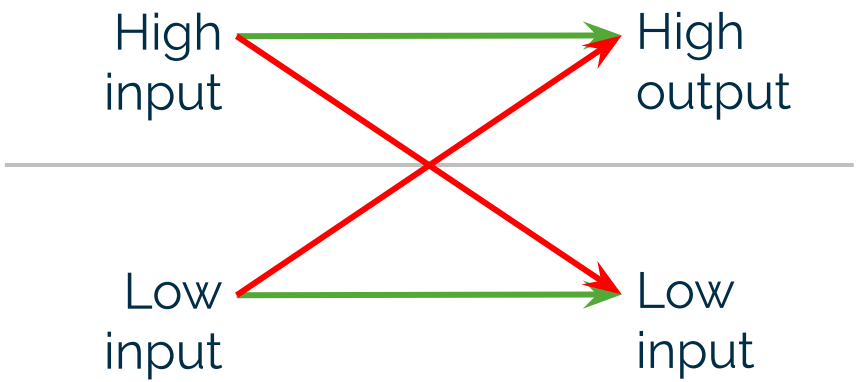
Integrity



ELIMITY

Multi-level access control

- You want both Bell-LaPadula and Biba
- However, this is not workable in practice
- => Refinement: Information flow control, taint tracking




```
var low, high
if check(high) then
  low := declassify(high)
```

Multi-level access control in the wild

- Core security feature of Windows Vista and newer
  - Complementary to discretionary access control
  - Control access to securable objects based on integrity level
  - Define the minimum integrity level required to access an object
- Isolate potentially untrustworthy contexts within the OS
  - Used by Google Chrome and Adobe Reader

svchost.exe		1.872 K	5.940 K	1844	Host Process for Windows S...	System	NT AUTHORITY\...
lsass.exe	0.15	4.032 K	11.496 K	484	Local Security Authority Proc...	System	NT AUTHORITY\...
lsm.exe	0.06	2.328 K	4.064 K	492	Local Session Manager Serv...	System	NT AUTHORITY\...
winlogon.exe	0.01	2.488 K	6.844 K	416	Windows Logon Application	System	NT AUTHORITY\...
explorer.exe	0.05	93.444 K	87.964 K	1416	Windows Explorer	Medium	Philippe-PC\Philippe
VBoxTray.exe	0.01	1.640 K	5.488 K	1180	VirtualBox Guest Additions Tr...	Medium	Philippe-PC\Philippe
POWERPNT.EXE	0.01	194.192 K	245.548 K	616	Microsoft PowerPoint	Medium	Philippe-PC\Philippe
WINWORD.EXE		44.144 K	91.400 K	3252	Microsoft Word	Medium	Philippe-PC\Philippe
procexp.exe		2.568 K	7.096 K	2932	Sysinternals Process Explorer	High	Philippe-PC\Philippe
procexp64.exe	0.99	14.356 K	25.040 K	2188	Sysinternals Process Explorer	High	Philippe-PC\Philippe
mspaint.exe		20.520 K	31.064 K	1112	Paint	Medium	Philippe-PC\Philippe
chrome.exe	0.05	44.944 K	72.500 K	236	Google Chrome	Medium	Philippe-PC\Philippe

Not just an academic exercise...

Pentagon investigates reporte

theguardian.com/world/2023/apr/07/pentagon-investigates-reported-leak-of-top-secret-ukraine-documents

Print subscriptions | Sign in | Search jobs | Search | International edition

Support the Guardian

Fearless, independent, reader-funded

Support us

The Guardian

News | Opinion | Sport | Culture | Lifestyle | More

World ▸ Europe US Americas Asia Australia Middle East Africa Inequality Global development

Ukraine

This article is more than 1 month old

Pentagon investigates reported leak of top-secret Ukraine documents

Classified papers said to contain details of military aid and battalion strengths before potential Ukrainian counteroffensive

Russia-Ukraine war - latest news updates

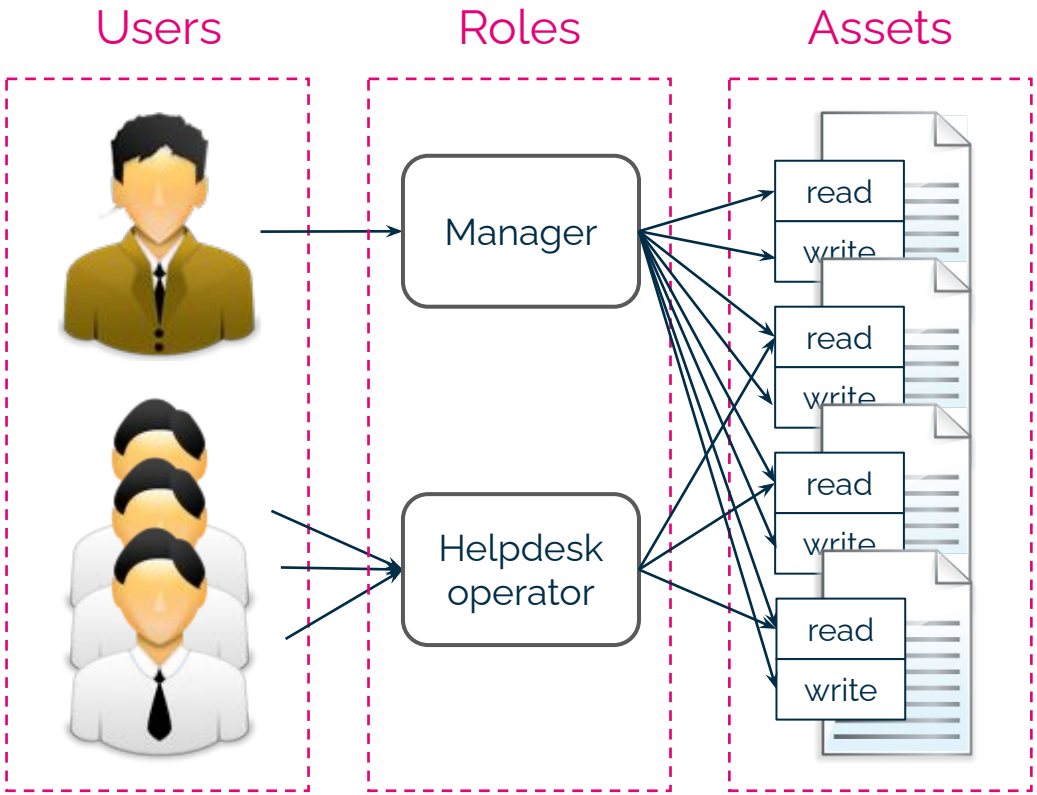
Lorenzo Tondo *and agencies*

Fri 7 Apr 2023 16.45 BST

https://www.theguardian.com

ELIMITY

Role-based access control (RBAC)



ELIMITY

Role-based access control (RBAC)

- Permissions assigned to roles, roles adopted by users
  - Goal: reduce large number of permissions to limited number of roles
  - Fits well onto the organizational structure of an enterprise
- Immense research field
  - Originated in research in 1992, NIST standard in 2004
  - Role hierarchies, role mining, administrative models, delegation, constraints, least privilege, static separation of duty through meta-rules, ...
- For app engineering: just group users in roles, don't make it too fancy :)

Outline

1. Introduction
2. **Deeper dive into access control**
   a. What is access control?
   b. Challenges
   c. Access control models
   d. How to implement
3. Deeper dive into IAM
4. How to IAM and access control relate?
5. Conclusion

Application-level access control

- Rules reason about the concepts in your application
- Add guard to code of your application
- The properties that you want:
  - Full mediation
  - Tamper proof
  - Verifiable

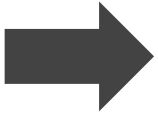
Option 1: encode guard and rules in app code

```
public  Document getDoc(docId) {
  Doc doc = db.getDoc(docId);
  if (! (“manager” in user.roles
      && doc.owner == user
      && 8h00 < now() < 17h00 )) {
    return null;
  } else {
    return doc;
  }
}
```

- \+ straightforward
- \+ you can encode almost anything
- \- no separation of concerns
- \- no modularity
  => hard for reviews
- \- what if rules change?
  - \- update application code
  - \- updates all over the place

Option 2: modularize

```
public  Document getDoc(docId) {
  Doc doc = db.getDoc(docId);
  if (! (“manager” in user.roles
      && doc.owner == user
      && 8h00 < now() < 17h00 )) {
    return null;
  } else {
    return doc;
  }
}
```



```
@authz(user, “read”, result)
public  Document getDoc(docId) {
  return db.getDoc(docId);
}
…
public boolean authz(
     user, action, resource) {
  if (!(“manager” in user.roles
          && …)) {
    return true;
  } else {
    return false;
  }
}
```

Option 2: modularize

- \+ more modularity: access control logic in 1 place
- \- no separation of concerns
- ± what if rules change?
  - \- update application code
  - \+ updates in one place

```
@authz(user, "read", result)
public  Document getDoc(docId) {
  return db.getDoc(docId);
}
…
public boolean authz(
     user, action, resource) {
  if (!("manager" in user.roles
          && …)) {
    return true;
  } else {
    return false;
  }
}
```

Option 2: modularize – Java Spring Security

In the controller:

```
@PreAuthorize("hasPermission(#doc, ‘view')")
public void getDocument(Document doc);
```

In the PermissionEvaluator:

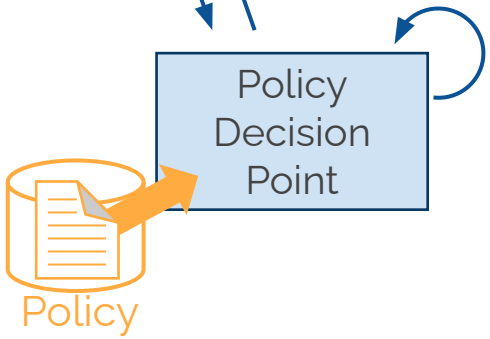
```
boolean hasPermission(Authentication a,
                      Object resource, String permission) {
  User user = SecurityUtil.getUserCredential();
  if(permission == “view” and ...) {
    return true;
  } else {
    return false;
  }
}
```

https://docs.spring.io/spring-security/site/docs/3.0.x/reference/el-access.html

Option 3: policy-based access control

```
@authz(user, "read", result)
public Document getDoc(docId) {
  return db.getDoc(docId);
}
…
public boolean authz(
     subject, action, resource) {
  if (! ("manager" in user.roles and …)) {
    return true;
  } else {
    return false;
  }}
```

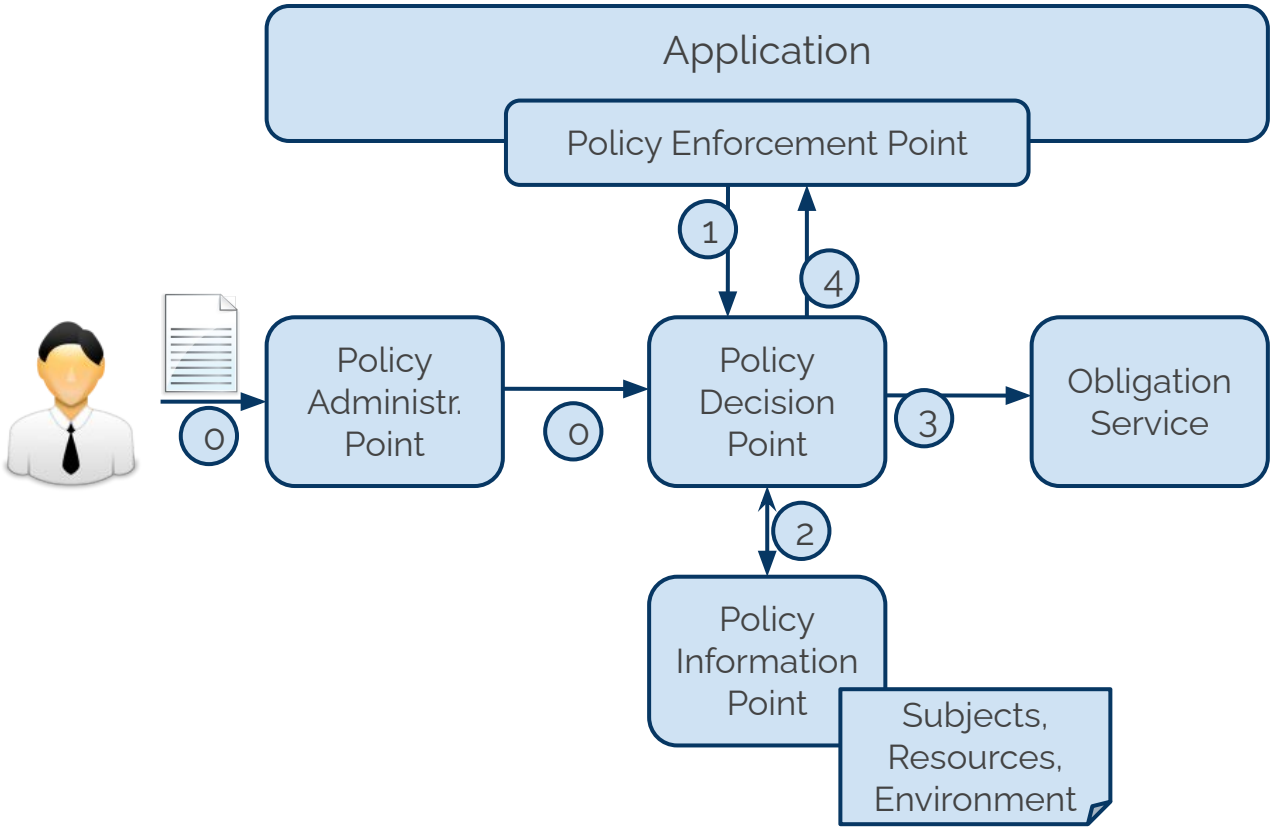
```
@authz(user, "read", result)
public Document getDoc(docId) {
  return db.getDoc(docId);
}
```



Option 3: policy-based access control

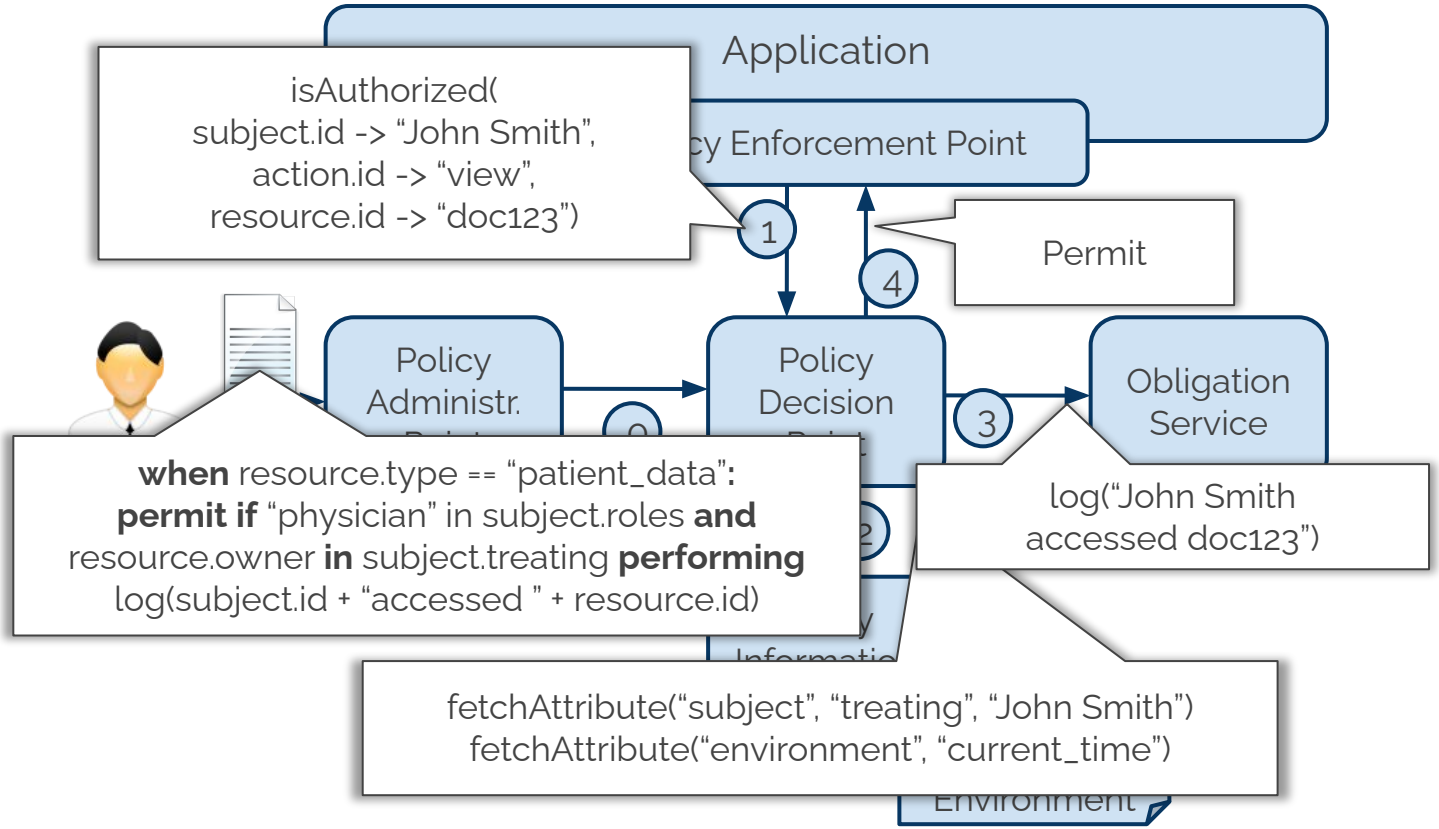
- Decouple access control rules from application code
  - Express access control rules in a format independent of your programming language
  - In application code: ask the generic question “can this subject perform this action on this resource”?
  - Policy evaluated by specialized component called the Policy Decision Point
  - If policy is stored in a file or a database: change policy at run-time

Reference architecture



ELIMITY

Reference architecture



ELIMITY

Advantages of PBAC

- More modularity: access control logic in 1 place
- Separation of concerns: policies can be written by non-developer
- What if rules change?
  - no updates in application code
  - updates in a single place
- Enables your access control policies to easily evolve with your organization
- Access rules are software artifacts => automated refinement, monitoring, validation, ...

PBAC in the wild: Amazon EC2

AWS | Services | Edit | Global | Support

Dashboard

Search IAM

Details

Groups

Users

Roles

Policies

Identity Providers

Account Settings

Credential Report

Encryption Keys

Description Policy to limit instance creation to specific regions and instance types. See https://forums.aws.amazon.com/thread.jspa?threadID=174503 .

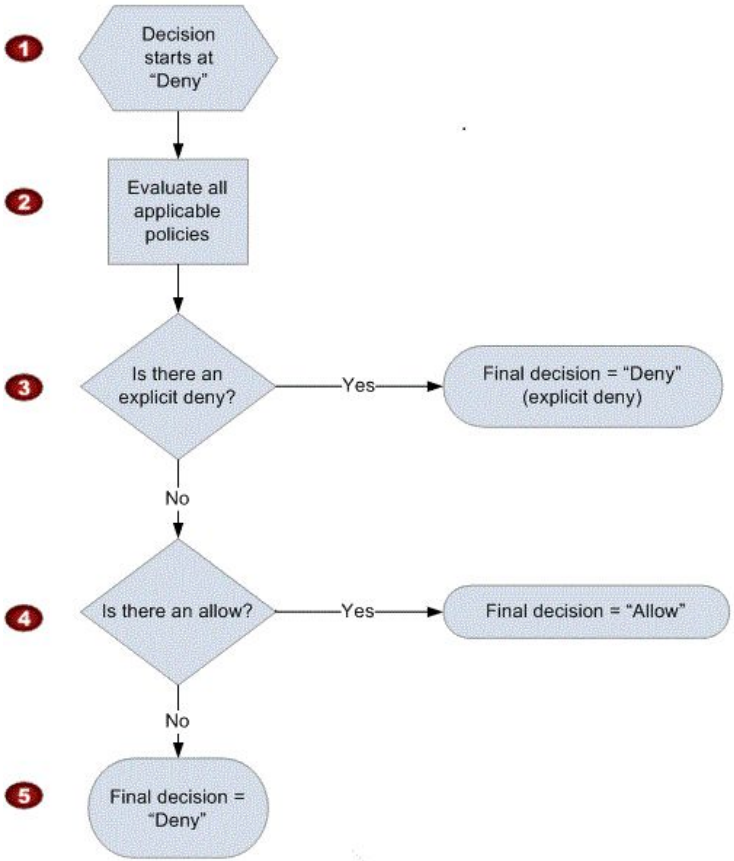
Policy Document | **Attached Entities** | **Policy Versions** | **Access Advisor**

Edit

```
14                ]
15            },
16            {
17                "Effect": "Allow",
18                "Action": "ec2:*",
19                "Resource": [
20                    "arn:aws:ec2:eu-west-1:*:*",
21                    "arn:aws:ec2:eu-west-1:*:security-group/*"
22                ],
23                "Condition": {
24                    "StringLikeIfExists": {
25                        "ec2:InstanceType": [
26                            "t2.micro",
27                            "t2.small",
28                            "t2.medium"
29                        ]
30                    }
31                }
32            },
33            {
34                "Effect": "Allow",
```

http://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_evaluation-logic.html

PBAC in the wild: Amazon EC2



http://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_evaluation-logic.html

PBAC in the wild: Amazon EC2

Policy Simulator

Amazon EC2 | 193 Action(s) se... | Select All | Deselect All | Reset Contexts | Clear Results | Run Simulation

Global Settings

Action Settings and Results [193 actions selected. 0 actions not simulated. 63 actions allowed. 130 actions denied.]

Service	Action	Resource Type	Simulation Resource	Permission
Amazon EC2	AcceptVpcPeeringConne...	vpc-peering-conn...	*	denied Implicitly denied (no matc...
Amazon EC2	ActivateLicense	not required	*	denied Implicitly denied (no matc...
Amazon EC2	AllocateAddress	not required	*	allowed 1 matching statements.
Amazon EC2	AssignPrivateIpAddresses	not required	*	denied Implicitly denied (no matc...
Amazon EC2	AssociateAddress	not required	*	allowed 1 matching statements.
Amazon EC2	AssociateDhcpOptions	not required	*	denied Implicitly denied (no matc...
Amazon EC2	AssociateRouteTable	not required	*	denied Implicitly denied (no matc...
Amazon EC2	AttachClassicLinkVpc	instance,security-...	*	denied Implicitly denied (no matc...
Amazon EC2	AttachInternetGateway	not required	*	denied Implicitly denied (no matc...
Amazon EC2	AttachNetworkInterface	not required	*	denied Implicitly denied (no matc...
Amazon EC2	AttachVolume	instance,volume	*	denied Implicitly denied (no matc...

ELIMITY

Policy languages

- A large number of domain-specific policy languages proposed in literature
  - E.g., SPL, Ponder, XACML, Cassandra, SecPAL, …
- Standard: XACML
  - Standardized by OASIS
  - v1.0 ratified in 2003, v3.0 in 2013
  - Attribute-based, tree-structured, obligations
  - XML format
  - Death by committee
- Platform-specific languages
  - E.g., Amazon AWS
- Hot new kid on the block: OPA

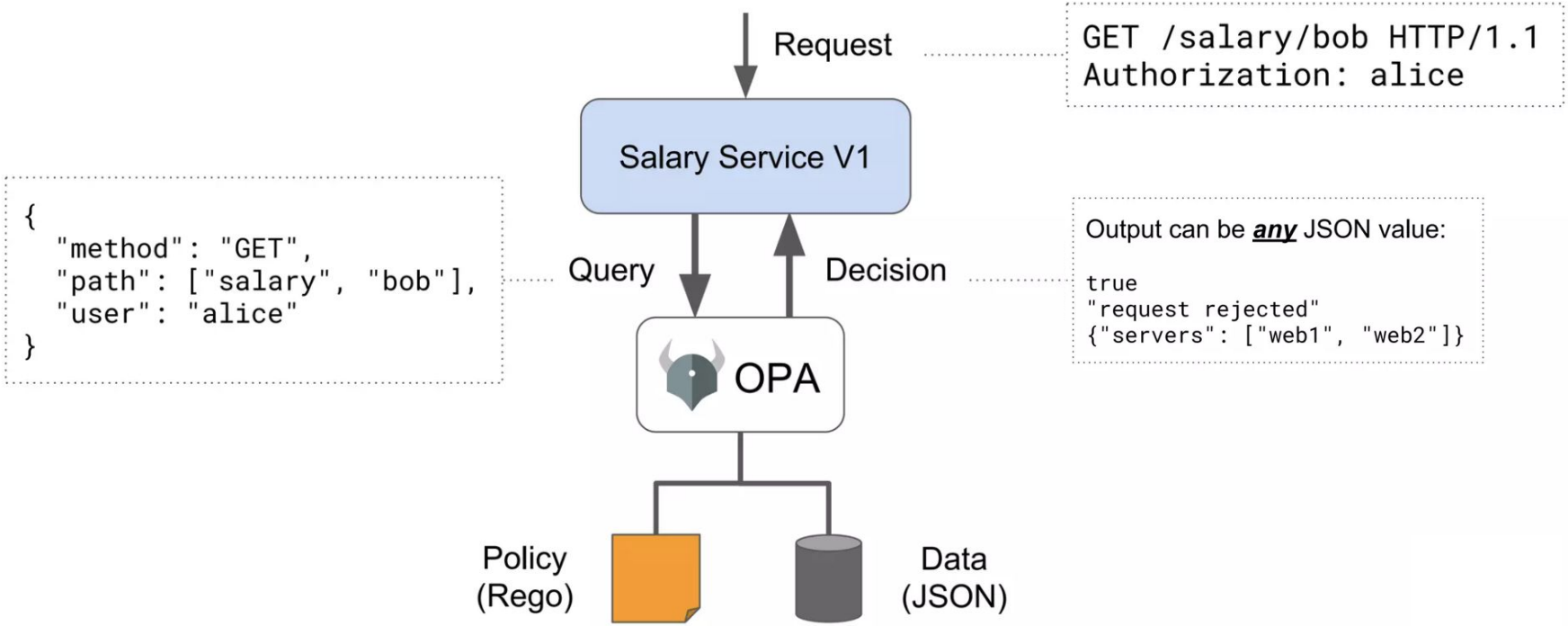
http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html

Policy languages: XACML

```
<Policy PolicyId="dynamic-separation-of-duty"
        RuleCombiningAlgId="deny-overrides">
  <Description>Dynamic separation of duty</Description>
  <Target>
    <Resources>
      <Resource>
        <ResourceMatch MatchId="string-equal">
          <AttributeValue DataType="string">doc123</AttributeValue>
          <ResourceAttributeDesignator AttributeId="resource:id" DataType="string"/>
        </ResourceMatch>
      </Resource>
    </Resources>
  </Target>
  <Rule RuleId="deny" Effect="Deny">
    <Description>Deny if viewed other doc</Description>
    <Condition>
      <Apply FunctionId="string-is-in">
        <AttributeValue DataType="string">doc456</AttributeValue>
        <SubjectAttributeDesignator AttributeId="subject:history" DataType="string"/>
      </Apply>
    </Condition>
  </Rule>
  <Rule RuleId="default-permit" Effect="Permit"> </Rule>
  <Obligations>
    <Obligation ObligationId="append-attribute" FulfillOn="Permit">
      <AttributeAssignment AttributeId="value" DataType="string">
        <SubjectAttributeDesignator AttributeId="resource:id" DataType="string"/>
      </AttributeAssignment>
      <AttributeAssignment AttributeId="attribute-id"
DataType="string">subject:history</AttributeAssignment>
    </Obligation>
  </Obligations>
</Policy>
```


ELIMITY

New kid on the block: Open Policy Agent (OPA)



Source: https://www.slideshare.net/TorinSandall/open-policy-agent-128970409?from_search=0

ELIMITY

New kid on the block: Open Policy Agent (OPA)

kubernetes Terraform docker	Admission Control	*"Restrict ingress hostnames for payments team."* *"Ensure container images come from corporate repo."*
envoy Istio Kong spring	API Authorization	*"Deny test scripts access to production services."* *"Allow analysts to access APIs serving anonymized data."*
Linux PAM	SSH & sudo	*"Only allow on-call engineers to SSH into production servers."*
ceph kafka MINIO	Data Protection	*"Trades exceeding $10M must be executed between 9AM and 5PM and require MFA."*
SQLite elastic	Data Filtering	*"Users can access files for past 6 months related to the region they licensed."*

Source: https://www.slideshare.net/TorinSandall/open-policy-agent-128970409?from_search=0

New kid on the block: Open Policy Agent (OPA)

Input

```
{
    "kind": "AdmissionReview",
    "request": {
        "kind": {
            "kind": "Pod",
            "version": "v1"
        },
        "object": {
            "metadata": {
                "name": "myapp"
            },
            "spec": {
                "containers": [
                    {
                        "image": "nginx",
                        "name": "nginx-frontend"
                    },
                    {
                        "image": "mysql",
                        "name": "mysql-backend"
                    }
                ]
            }
        }
    }
```

Policy

```
package kubernetes.validating

deny[msg] {
    not input.request.object.metadata.labels.costcenter
    msg := "Every resource must have a costcenter label"
}
```

Output

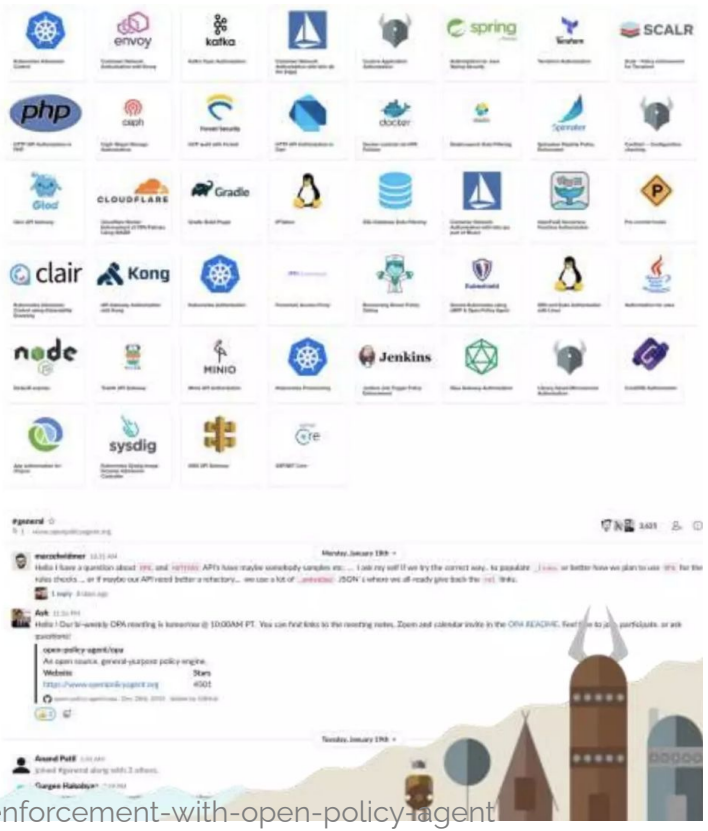
```
{
    "deny": [
        "Every resource must have a costcenter label"
    ]
}
```

Source: https://www.slideshare.net/LibbySchulze/cloud-native-policy-enforcement-with-open-policy-agent

New kid on the block: Open Policy Agent (OPA)

Vibrant community

- 160 contributors
- 50+ integrations
- 4500+ Github Stars
- 3600+ Slack users
- 30+ million Docker image pulls
- Ecosystem including Conftest, Gatekeeper, VS Code and IntelliJ editor plugins.



Source: https://www.slideshare.net/LibbySchulze/cloud-native-policy-enforcement-with-open-policy-agent

ELIMITY

Advantages of PBAC

- More modularity: access control logic in 1 place
- Separation of concerns: policies can be written by non-developer
- What if rules c
  - no updates in a ion code
  - updates in a single place
- Enables your access control policies to easily evolve with your organization
- Enables centralizing policies, explicitly managing policies across your organization, refining business policies, …

Ideally

ELIMITY

Not all rainbows and unicorns

- Very interesting technology, great vision to work towards
- But, policy-based access control is (still) very hard in practice:
  - Different way of coding
  - Policy languages are not self-explanatory
  - Requires your customers to have processes for managing policies within their org
  - The trusted computing base of your application grows significantly
  - Plus, from research experience: inherently hard to decouple authorization logic from an application because these rules should still say something about *this* application

My recommendation: definitely modularize authorization in your application code (option 2), but only apply PBAC if you really need the flexibility, e.g., OPA in microservices or you're building the next AWS.

Access control: summary

- Access control is a key part of protecting the data in your application
- Advice to avoid access control vulnerabilities:
  - Full mediation + deny by default
  - Modularize access control in your code
  - Know the different access control models in research, but keep the access control model of your application as simple as possible (KISS)

Outline

1. Introduction
2. Deeper dive into access control
3. **Deeper dive into IAM**
   a. The 4 disciplines of IAM
   b. RBAC & ABAC
4. How to IAM and access control relate?
5. Conclusion

What is IAM?

Identity & Access Management (IAM)
encompasses all processes used by an organization
to ensure that everyone can access the data they need
and only the data that they need.

What is IAM?

ELIMITY

The 4 disciplines of IAM

1. Authentication

Minimize the chances of credential theft

SSO, MFA, provisioning, ...

Most technical discipline

2. IGA

Identity governance & administration

Manage the lifecycle of the identities of your employees and their accesses

Joiner/mover/leaver

Access requests & approvals

Access reviews & revocations

Most complex discipline, goes far beyond IT

3. PAM

Privileged access management

Govern the highly-privileged accounts (admins) in your IT systems

Password vaulting

Password rotation

Session management & monitoring

Requires your admins to change their way of working = like herding cats

4. CIAM

Consumer IAM

IAM for external identities (customers)

Mainly relevant if you are a software provider

Main challenge is scale

Limited security impact

ELIMITY

The 4 disciplines of IAM

1. Authentication

Minimize the chances of credential theft

SSO, MFA, provisioning, ...

Most technical discipline

2. IGA

Identity governance & administration

Manage the lifecycle of the identities of your employees and their accesses

Joiner/mover/leaver

Access requests & approvals

Access reviews & revocations

Most complex discipline, goes far beyond IT

3. PAM

Privileged access management

Govern the highly-privileged accounts (admins) in your IT systems

Password vaulting

Password rotation

Session management & monitoring

Requires your admins to change their way of working = like herding cats

4. CIAM

Consumer IAM

IAM for external identities (customers)

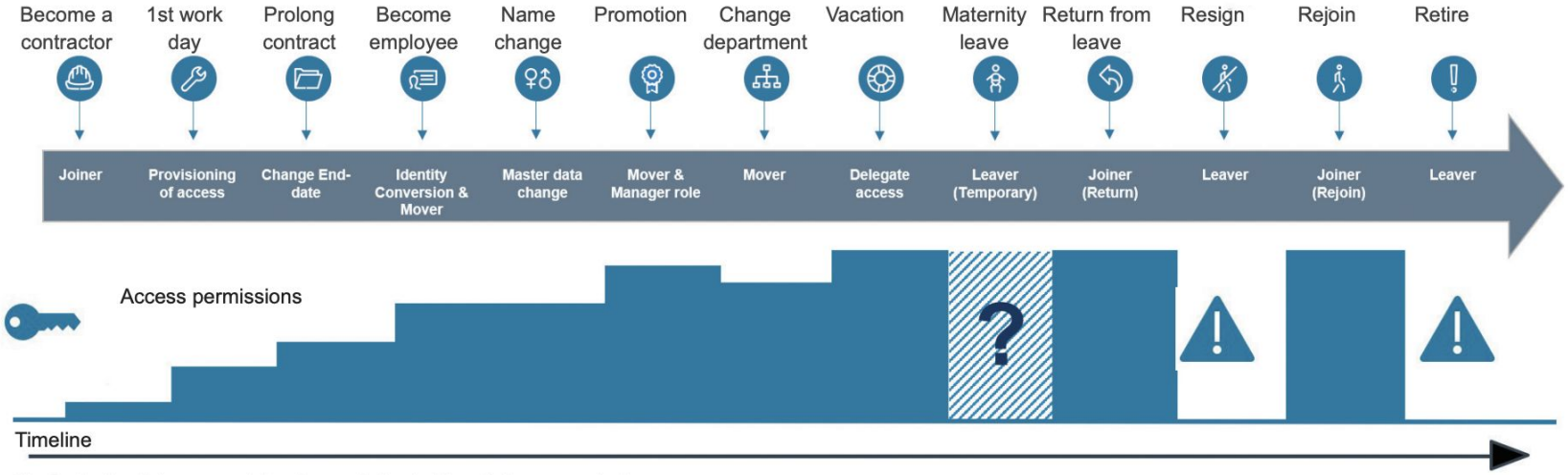
Mainly relevant if you are a software provider

Main challenge is scale

Limited security impact

ELIMITY

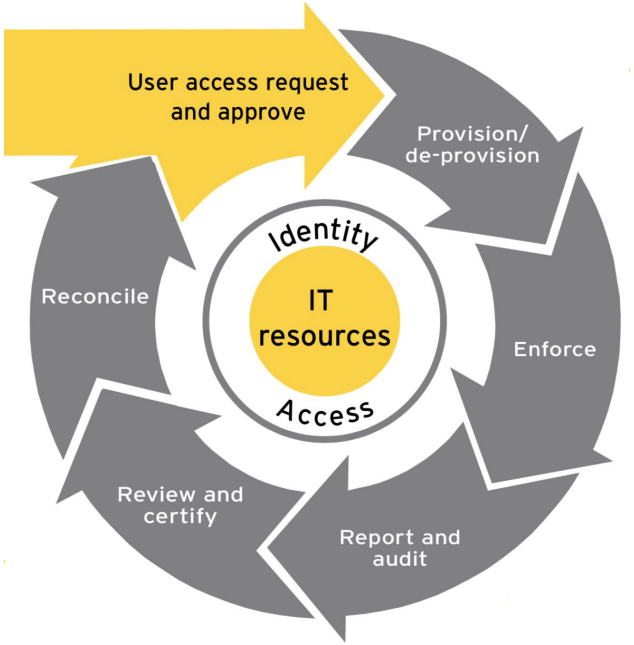
The 4 disciplines of IAM - IGA



Source: Omada IdentityPROCESS+, Version 2.0

ELIMITY

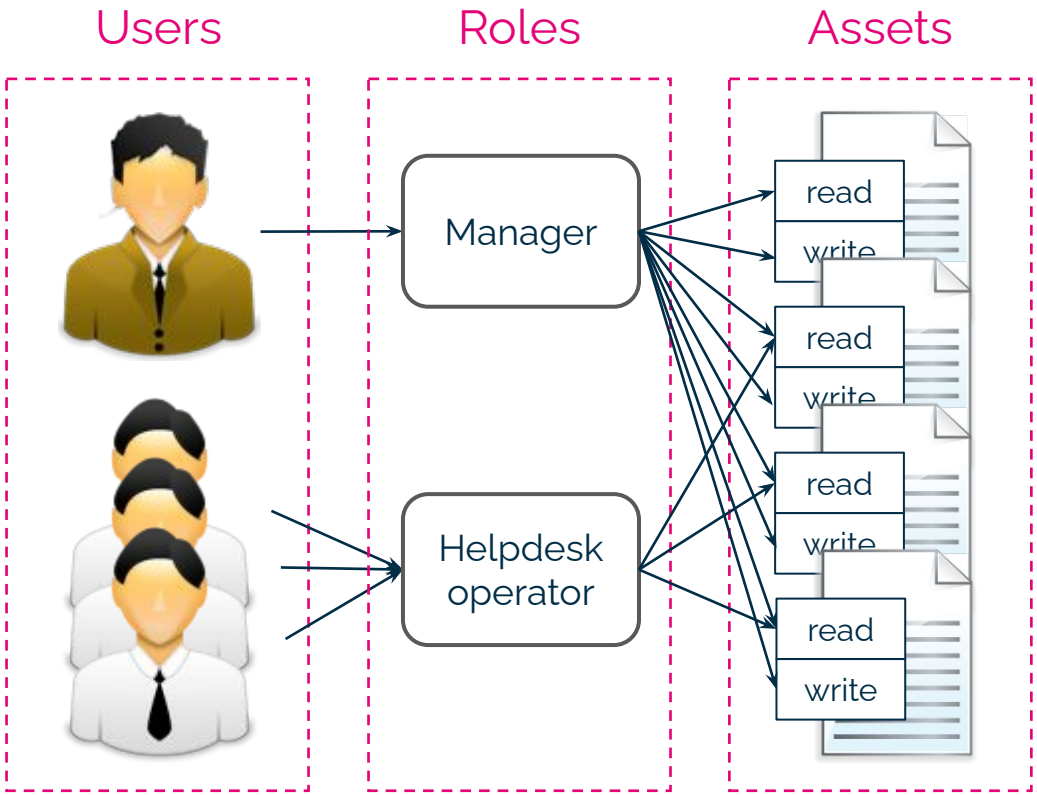
The 4 disciplines of IAM - IGA



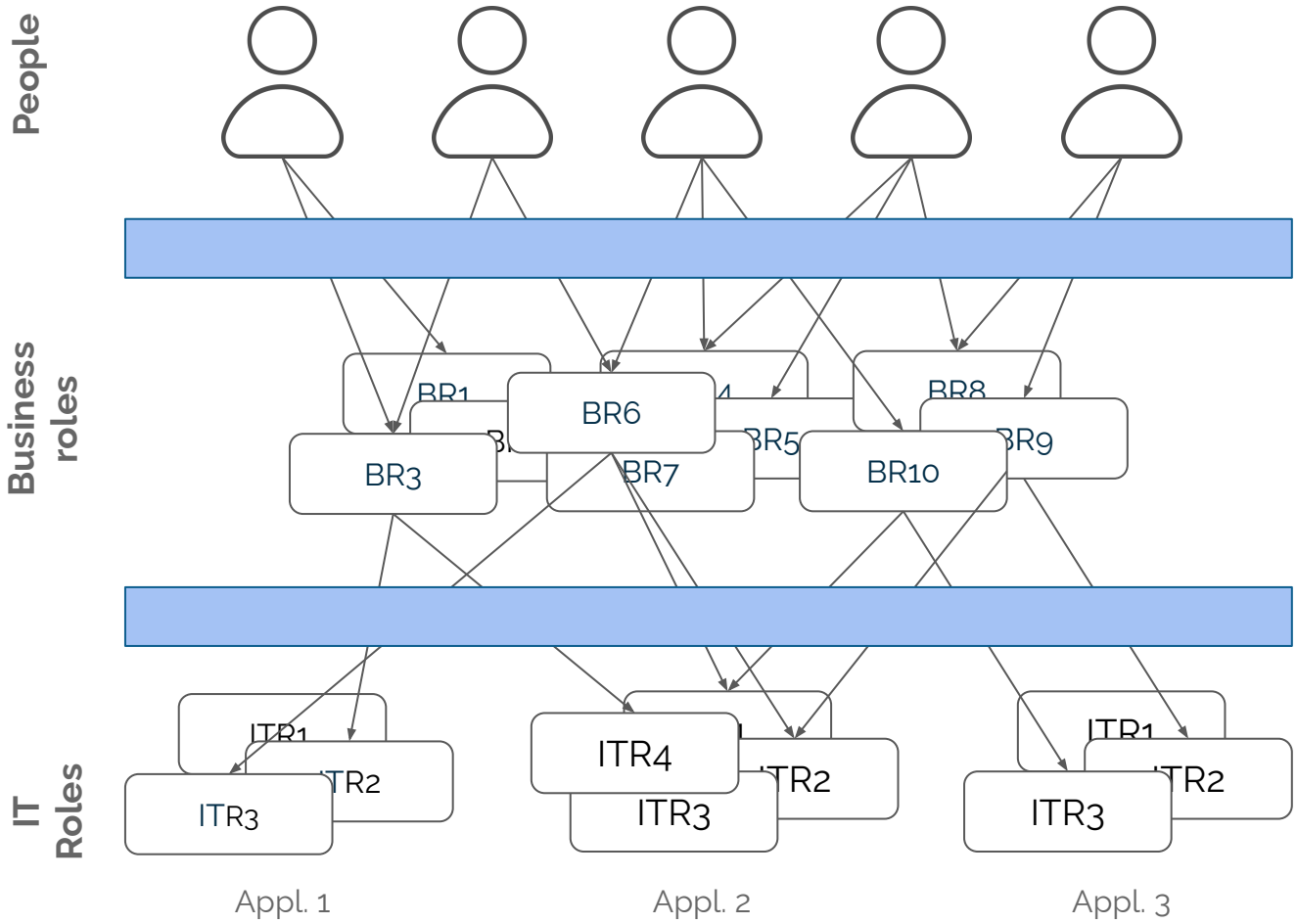
Source: Insights on Governance, Risk and Compliance:
Identity and access management - Beyond compliance, EY, May 2013

ELIMITY

Role-based access control (RBAC)

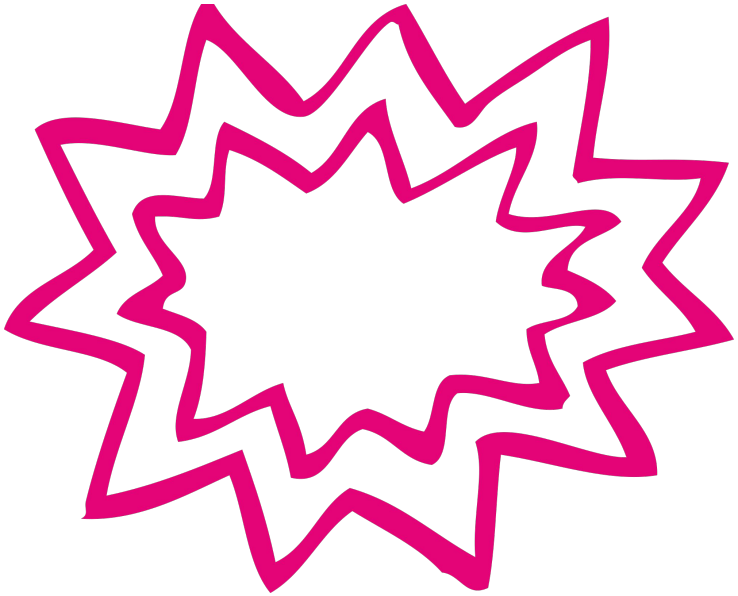


ELIMITY

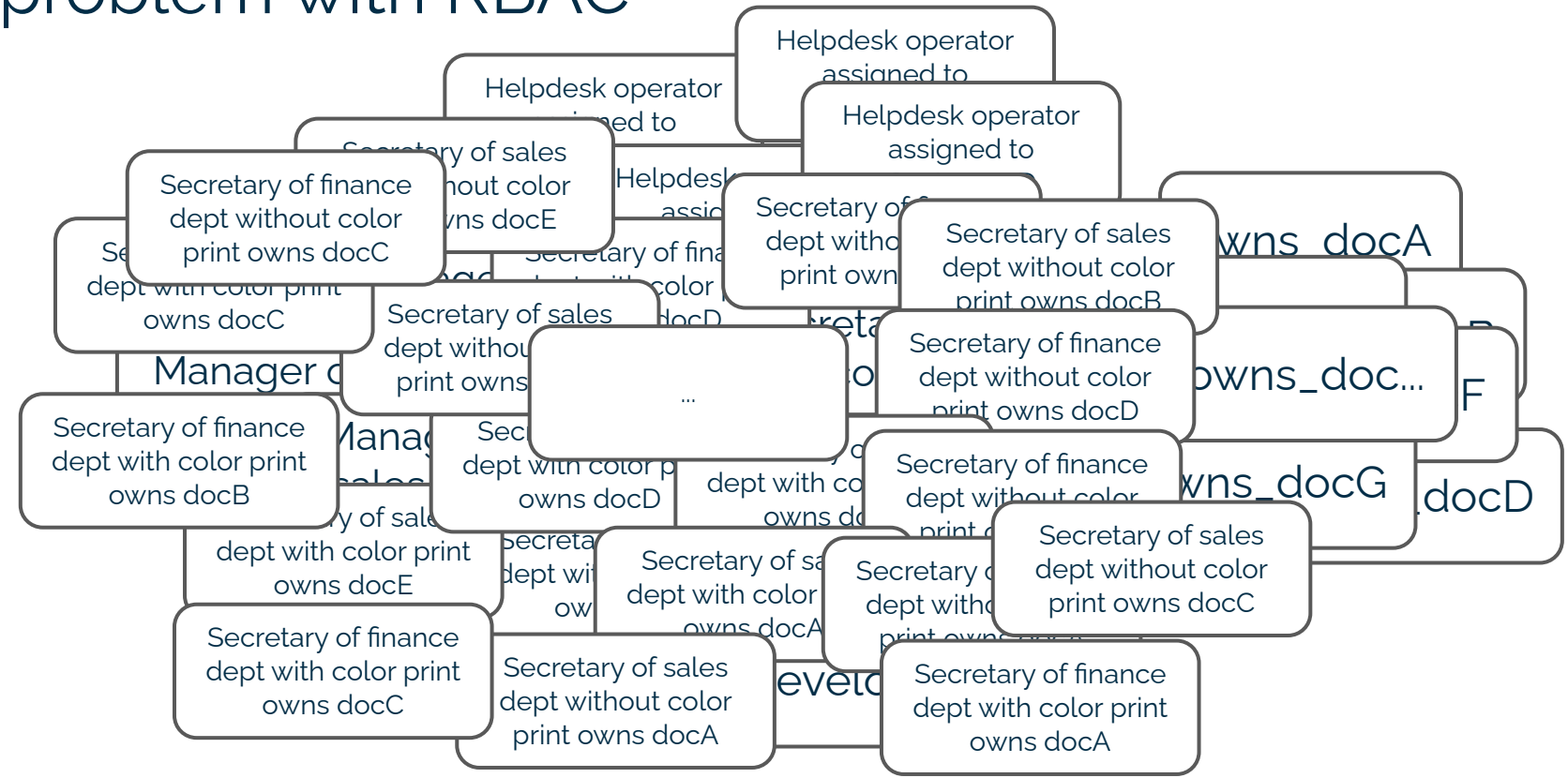

People
Business roles
IT Roles
BR6
BR3
BR7
BR5
BR10
BR9
ITR4
ITR3
ITR2
ITR3
ITR2
ITR3
Appl. 1
Appl. 2
Appl. 3

ELIMITY

The problem with RBAC



The problem with RBAC

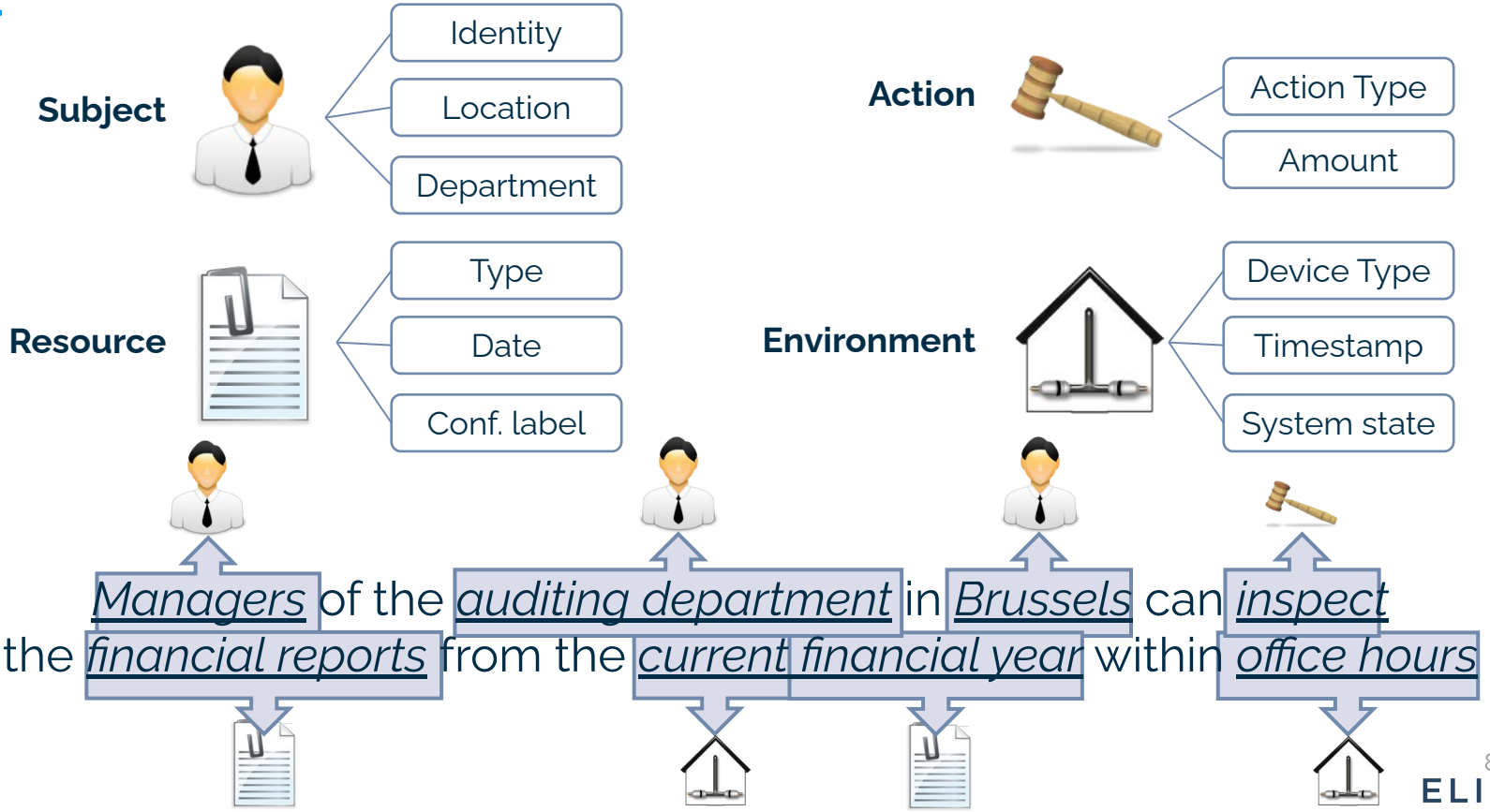


ELIMITY

Role-based access control (RBAC)

- Major disadvantage: role explosion
- Reasons:
  - Roles cannot express ownership
    - Requires roles like “owns_docA”, “owns_docB”, etc
  - Reality is too fine-grained
    - Often small differences between different persons *in the same job*, leading to yet another role (e.g., “secretary_with_colorprint”)
  - Cross-product of multiple hierarchies
    - E.g., “sales_manager_for_belgium_with_colorprint_owns_docA”
- As a result:
  - Hard to get right, moving target
  - Large overhead at any decently-sized company

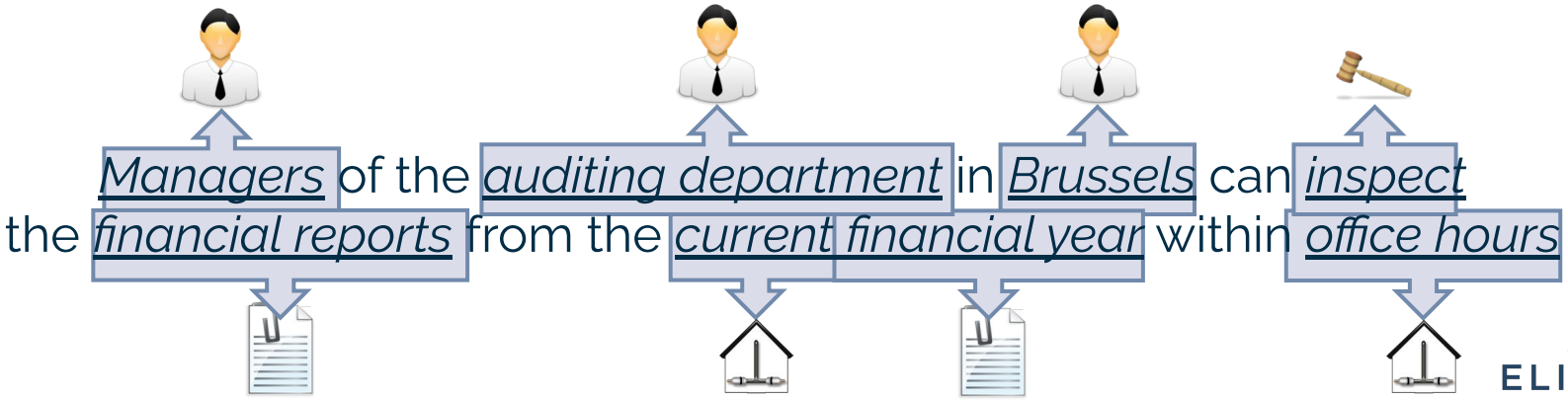
Attribute-based Access Control (ABAC)



ELIMITY

Attribute-based Access Control (ABAC)

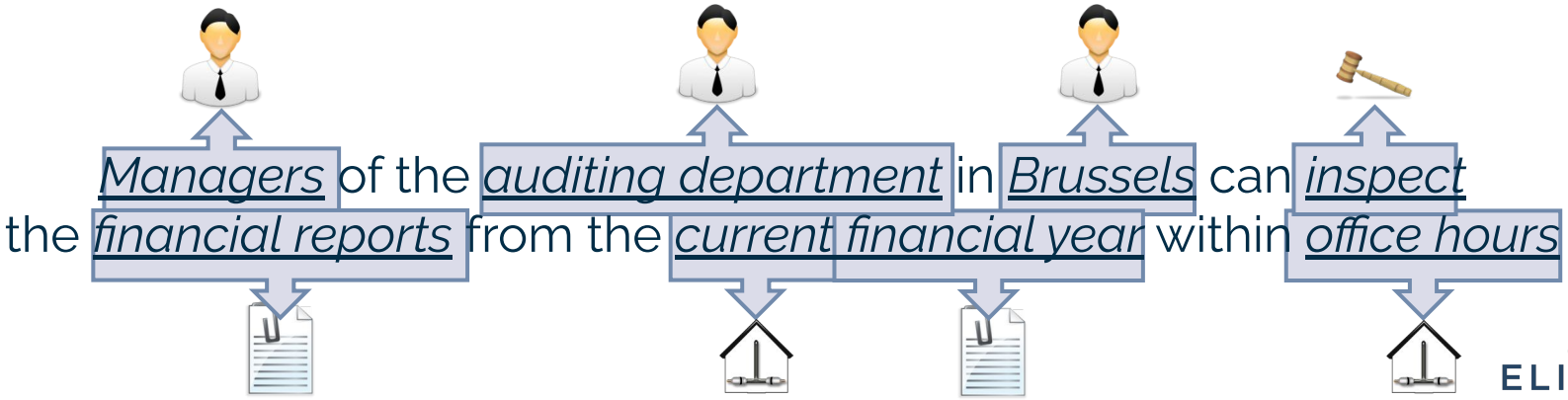
permit if
"manager" in subject.roles and subject.department == "auditing"
and subject.location == "Brussels" and action == "inspect"
and resource.type == "financial report"
and resource.year == environment.current_year
and 8h00 < environment.time < 17h00



ELIMITY

Attribute-based Access Control (ABAC)

1. fine-grained access control
2. context-aware access control
3. dynamic access control

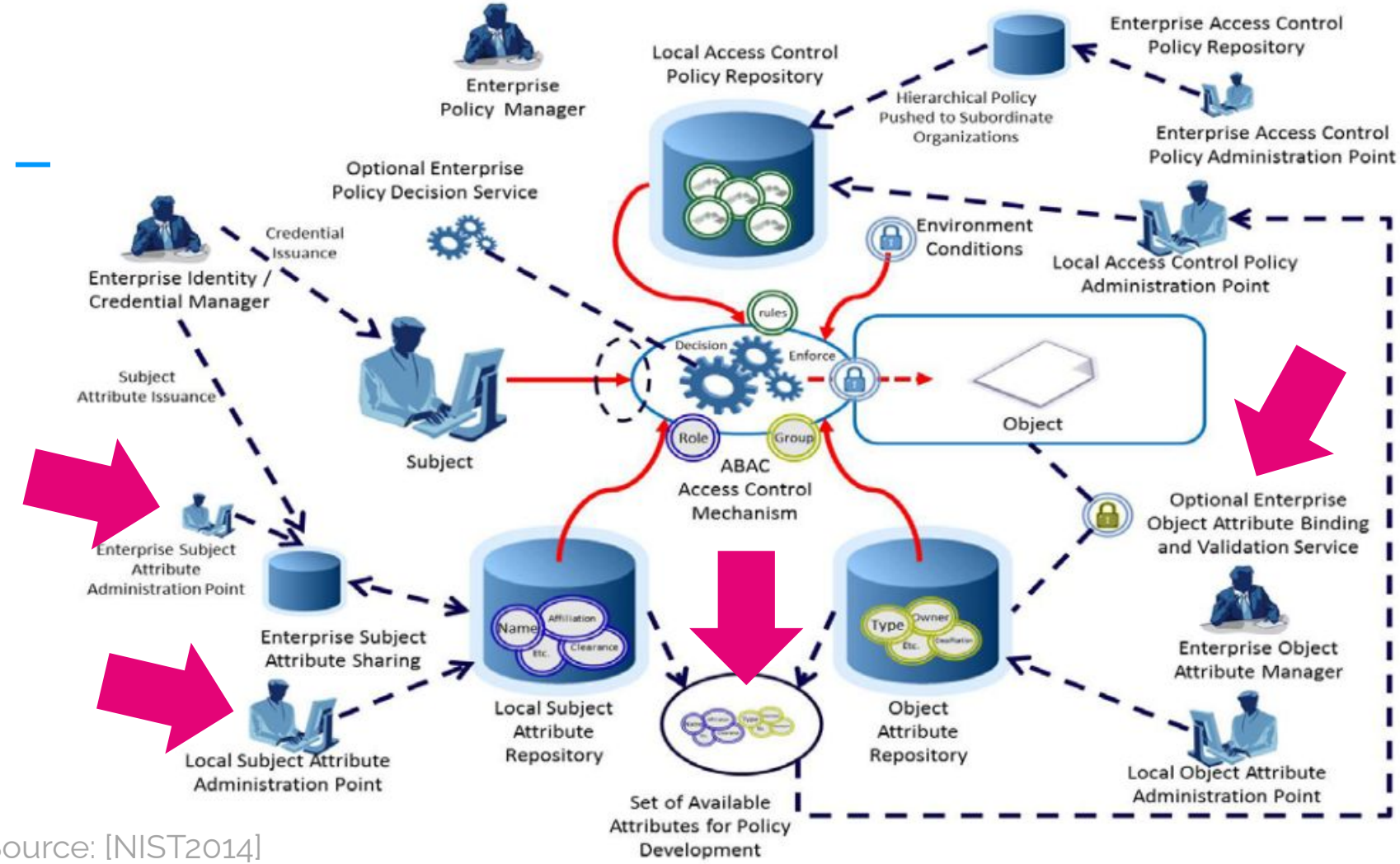


ELIMITY

Attribute-based Access Control (ABAC)

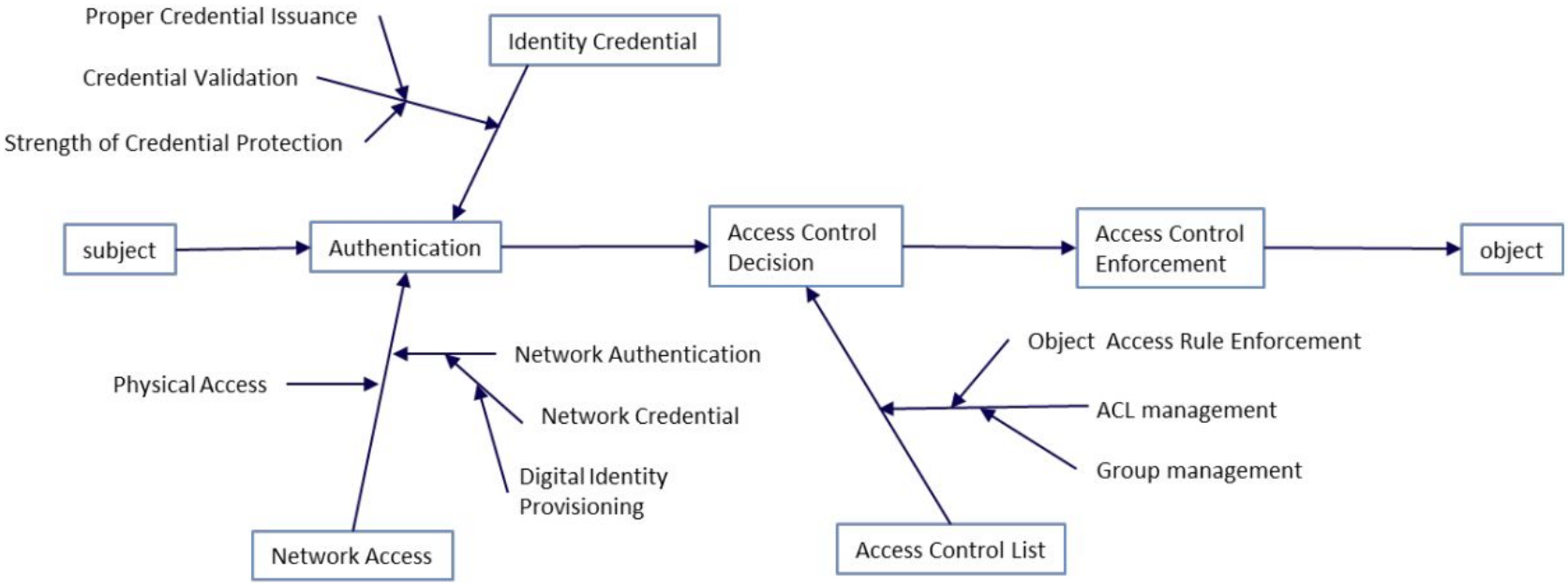
- Access decisions are made based on attributes
  - Attributes are key-value properties of the subject, the resource, the action or the environment
  - Results into dynamic and context-aware access control
- Attributes can express many different access control concepts
  - Permissions, roles, groups, departments, time, location, ownership, domain-specific ownership, …
- Together with PBAC, this is sometimes regarded as the holy grail of access control. However…

Not all rainbows and unicorns



Source: [NIST2014]

Not all rainbows and unicorns

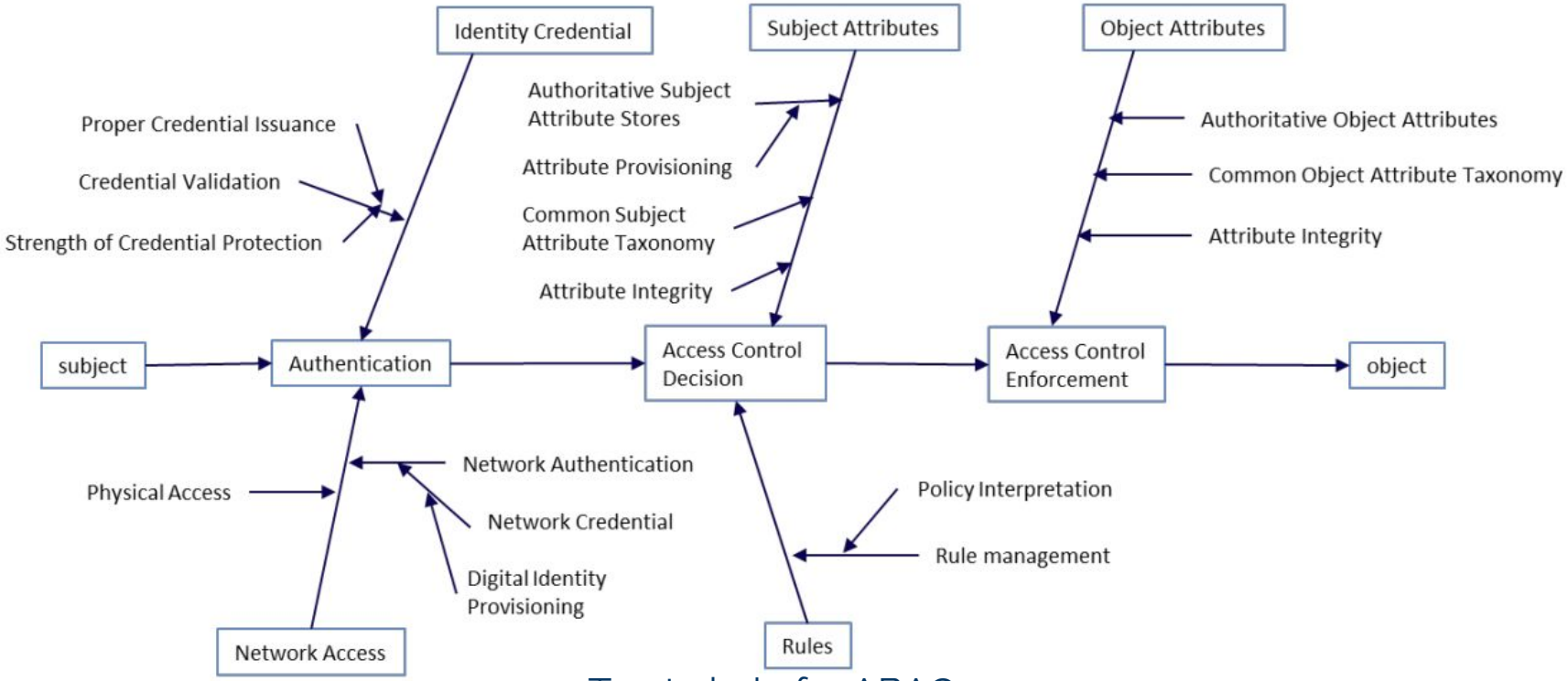


Trust chain for Access Control Lists

Source: [NIST2014]

ELIMITY

Not all rainbows and unicorns



Trust chain for ABAC

Source: [NIST2014]

ELIMITY

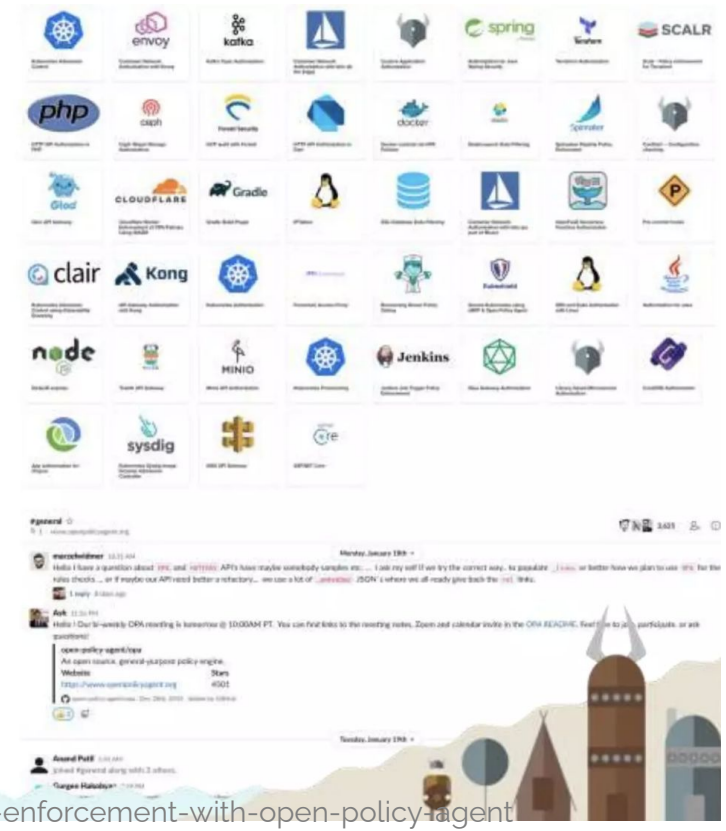
Not all rainbows and unicorns

"Enterprise ABAC carries with it significant development, implementation, and operations costs as well as a paradigm shift in the way enterprise objects are shared and protected." -- NIST

New kid on the block: Open Policy Agent (OPA)

Vibrant community

- 160 contributors
- 50+ integrations
- 4500+ Github Stars
- 3600+ Slack users
- 30+ million Docker image pulls
- Ecosystem including Conftest, Gatekeeper, VS Code and IntelliJ editor plugins.



Source: https://www.slideshare.net/LibbySchulze/cloud-native-policy-enforcement-with-open-policy-agent

ELIMITY

Outline

1. Introduction
2. Deeper dive into access control
3. Deeper dive into IAM
4. **How to IAM and access control relate?**
5. Conclusion

The 4 disciplines of IAM

1. Authentication

Minimize the chances of credential theft

SSO, MFA, provisioning, ...

Most technical discipline

2. IGA

Identity governance & administration

Manage the lifecycle of the identities of your employees and their accesses

Joiner/mover/leaver

Access requests & approvals

Access reviews & revocations

Most complex discipline, goes far beyond IT

3. PAM

Privileged access management

Govern the highly-privileged accounts (admins) in your IT systems

Password vaulting

Password rotation

Session management & monitoring

Requires your admins to change their way of working = like herding cats

4. CIAM

Consumer IAM

IAM for external identities (customers)

Mainly relevant if you are a software provider

Main challenge is scale

Limited security impact

ELIMITY

The 4 disciplines of IAM

1. Authentication

Minimize the chances of credential theft

SSO, MFA, provisioning, ...

Most technical discipline

2. IGA

Identity governance & administration

Manage the lifecycle of the identities of your employees and their accesses

Joiner/mover/leaver

Access requests & approvals

Access reviews & revocations

Most complex discipline, goes far beyond IT

3. PAM

Privileged access management

Govern the highly-privileged accounts (admins) in your IT systems

Password vaulting

Password rotation

Session management & monitoring

Requires your admins to change their way of working = like herding cats

4. CIAM

Consumer IAM

IAM for external identities (customers)

Mainly relevant if you are a software provider

Main challenge is scale

Limited security impact

ELIMITY

How does this relate to access control?

1. Authentication

Minimize the chances of credential theft

SSO, MFA, provisioning, ...

Most technical discipline

2. IGA

Identity governance & administration

Manage the lifecycle of the identities of your employees and their accesses

Joiner/mover/leaver

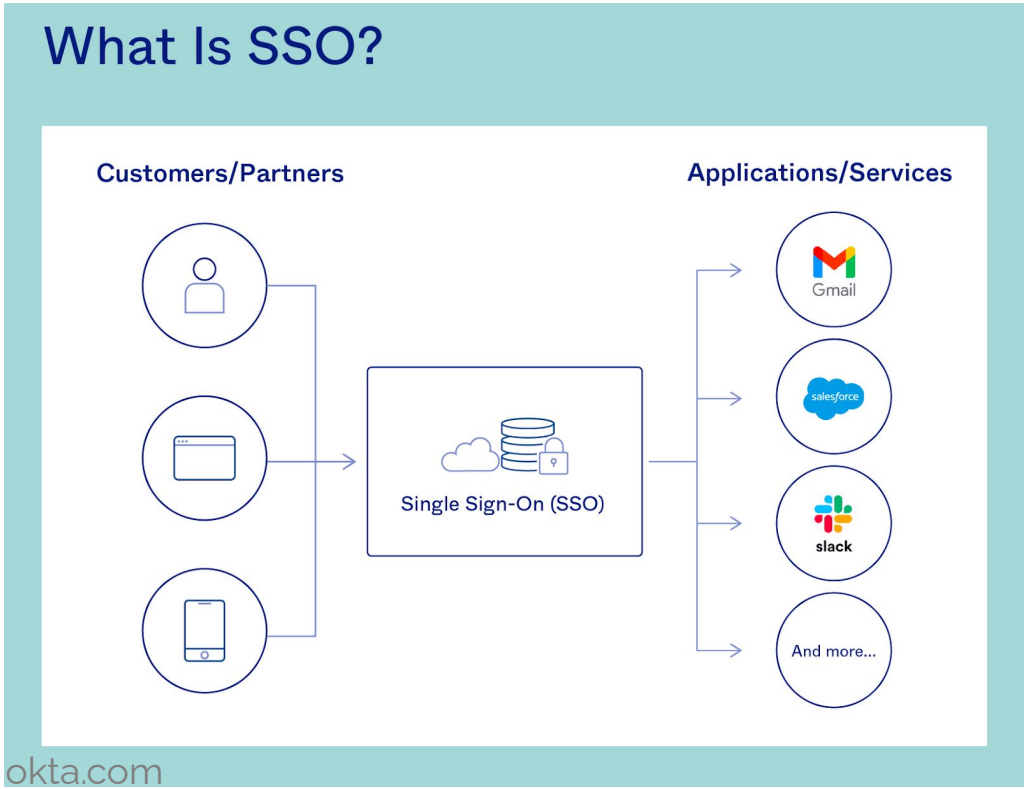
Access requests & approvals

Access reviews & revocations

Most complex discipline, goes far beyond IT

If we know that this is crucial to our customer's security, let's make sure to help them the best we can do!

Authentication: managing the chaos



Source: okta.com

For applications:

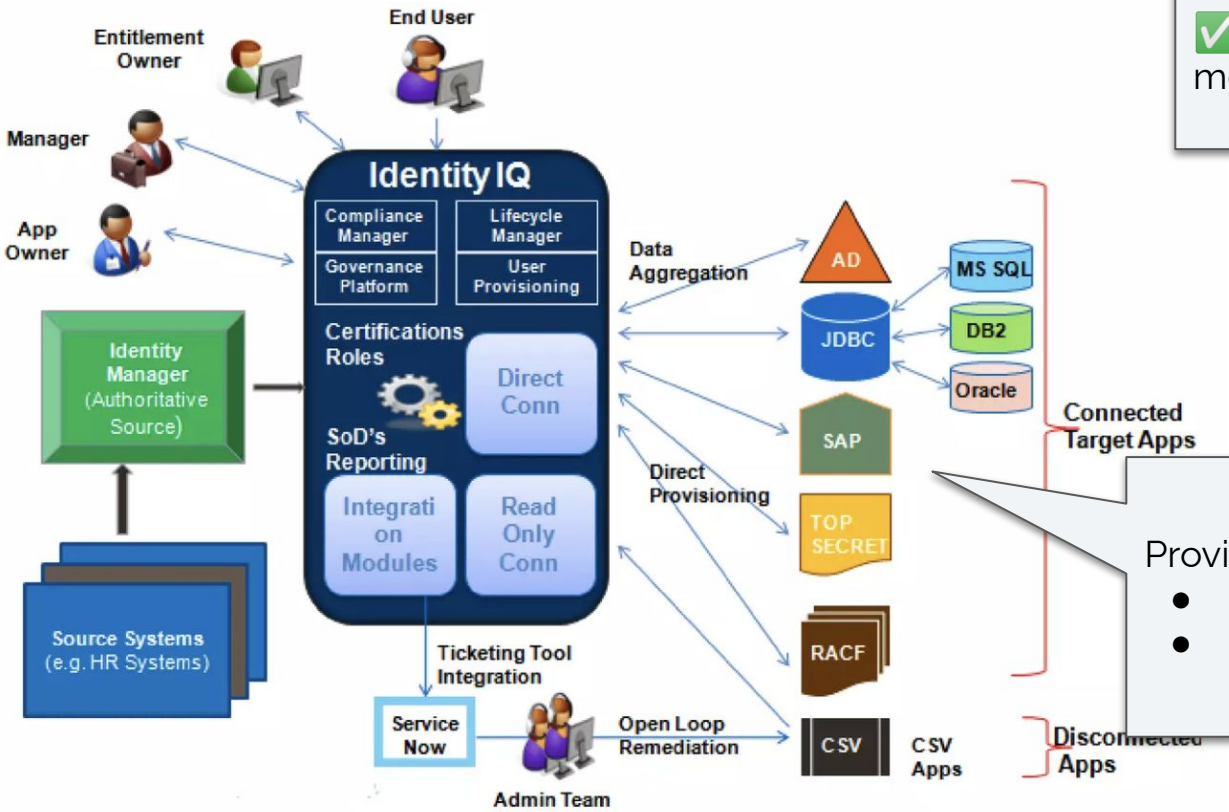
✅ Support SSO: AD or OIDC

ELIMITY

IGA: managing the chaos

For applications:

✅ Support external user mgmt: AD or SCIM



Provisioning:

- AD integration if on prem
- SCIM if SaaS

Source: https://www.slideshare.net/proferyk/identity-access-management-for-securing-devops

ELIMITY

SCIM = Id Mgmt + REST

- REST is just an architectural pattern
- SCIM defines endpoints for identity mgmt:
  - Standard definitions for User and Group
  - All expressed in JSON
  - Standard operations
    - Create, read, update, delete, search, partial update, bulk
  - Extensibility

Example: Slack SCIM API

Endpoints

The SCIM API is RESTful and the endpoint URLs are different than other Slack API endpoints.

Endpoint	Description
`GET /ServiceProviderConfigs`	Returns Slack's configuration details for our SCIM API
`GET /Schemas/Users`	Returns Slack's configuration details for how users are formatted
`GET /Schemas/Groups`	Returns Slack's configuration details for how groups are formatted
`GET /Users`	Returns a paginated list of users
`GET /Users/<id>`	Retrieves a single user resource
`POST /Users`	Creates a user
`PATCH /Users/<id>`	Updates an existing user resource, overwriting specified values
`PUT /Users/<id>`	Updates an existing user resource, overwriting all values
`DELETE /Users/<id>`	Sets a Slack user to deactivated
`GET /Groups/`	Returns a paginated list of groups
`GET /Groups/<id>`	Retrieves a single group resource
`POST /Groups`	Creates a new group
`PATCH /Groups/<id>`	Updates an existing group resource
`PUT /Groups/<id>`	Updates an existing group resource, overwriting all values
`DELETE /Groups/<id>`	Permanently removes a group

Example: request to retrieve user

```
GET /scim/v2/Users/23a35c27-23d3-4c03-b4c5-6443c09e7173 HTTP/1.1
User-Agent: Okta SCIM Client 1.0.0
Authorization: <Authorization credentials>
```

Example: request to retrieve user

```
HTTP/1.1 200 OK
Date: Tue, 10 Sep 2019 03:46:53 GMT
Content-Type: text/json;charset=UTF-8

{
    "schemas": ["urn:ietf:params:scim:schemas:core:2.0:User"],
    "id": "23a35c27-23d3-4c03-b4c5-6443c09e7173",
    "userName": "test.user@okta.local",
    "name": { "givenName": "Test", ... },
    "active": true,
    "emails": [{ "primary": true, "value": "test.user@okta.local", "type": "work", ... }],
    "groups": [],
    "meta": {
        "resourceType": "User"
    }
}
```

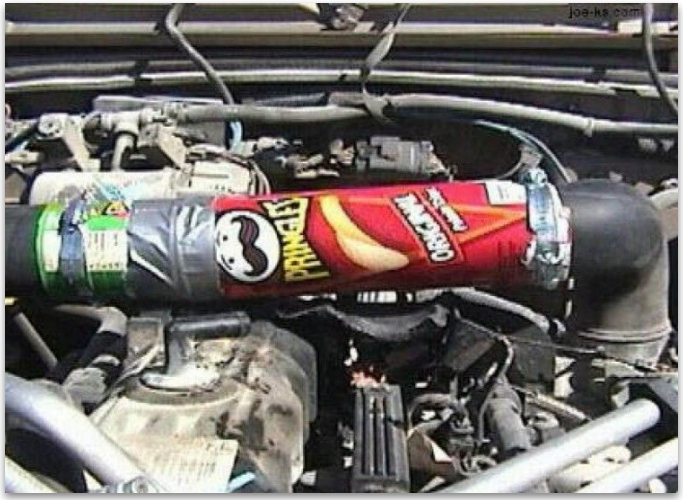
Self-describing payload

Single-valued attributes
Many different data types

Complex attributes

ELIMITY

Sound nice in theory, but...



- Many access control models don't fit in lists of groups
  - E.g., ownership, roles vs profiles, ...
  - Leading to many implicit assumptions behind the API
  - Leading to strange behavior
  - Leading to large overhead to write these types of integrations
- SCIM v2: extensible schema's for core objects, new types of objects, fully customizable, but...
  - Do the clients properly support this? Can IGA suites work with this?
  - Note that SCIM proxies are actually offered commercially these days, SCIM is here to stay
- I would still recommend adding a SCIM API to your application, still the best that we can do
  - The result will also be better if you have kept your access control model simple

Outline

1. Introduction
2. Deeper dive into access control
3. Deeper dive into IAM
4. How to IAM and access control relate?
5. **Conclusion**

Conclusion

- Access control: essential part of an application's security
- IAM: essential part of an organisation's security
- However: both remain challenging and no silver bullets
- The goal of this presentation: give you structured insights in what access control and IAM are, so you are better prepared to handle these topics in practice
- Common thread: don't over-complicate things :-)

Thank you

Maarten Decat

Helping companies get in control of who can access what

maarten@elimity.com +32 472 599 055

Personal 30min follow-up:

https://calendly.com/maarten-decat/30min

More reading

IDPro: community of identity experts with vast body of knowledge

How to prove that you are in control

How to build the perfect risk cockpit for Active Directory

For more guides, visit:
www.elimity.com/resources