Plan de Projet : Transitive Trust Chain Abuse in Modern Cloud IAM

Introduction et Contexte Stratégique

Ce document présente le plan de travail détaillé pour le projet de recherche et développement intitulé provisoirement : **"Transitive Trust Chain Abuse in Modern Cloud IAM – Formalisation d'une classe d'attaque et outil de cartographie/détection multi-principal"**.

L'objectif stratégique de ce projet est double :

1. **Reconnaissance Académique :** Contribuer à la littérature en cybersécurité par une formalisation rigoureuse de la classe d'attaque CTI (Chaîne de Confiance Transitive) dans les environnements Cloud modernes (IAM).
2. **Impact Industriel :** Fournir à la communauté open-source et aux équipes de sécurité (SOC, Cloud Security Architects) un outil pratique, reproductible et sans coût, capable de cartographier et de détecter ces chemins d'escalade de privilèges.

Le plan est structuré sur une période de **12 semaines** et est conçu pour maximiser l'efficacité en solo, en s'appuyant exclusivement sur des technologies open-source et des environnements de simulation (Docker, LocalStack) afin de respecter la contrainte de budget nul.

Objectif Global du Projet

Formaliser la classe d'attaque par abus de chaînes de confiance transitives dans les systèmes IAM Cloud (simulés) et développer un outil open-source de cartographie et de détection multi-principal.

Phases Détaillées du Projet (Roadmap 12 Semaines)

Le projet est découpé en six phases séquentielles, chacune d'une durée de deux semaines, garantissant une progression logique de la théorie à la pratique, puis à la publication.

Phase	Durée (Semaines)	Objectifs Précis	Tâches Clés (MVP en gras)	Livrables Attendus
P1. Recherche & Modélisation Formelle	1-2	Établir le cadre théorique et la formalisation de l'attaque.	**1.1. Revue de littérature approfondie** (PMapper,	Document de Modélisation Formelle (CTI-Model.md).

			BloodHound, CloudMapper). **1.2. Définition formelle de la "Confiance Transitive IAM" (CTI)**. **1.3. Modélisation de l'IAM en Graphe** (Identités, Permissions, Relations de Confiance).	Schéma de Graphe IAM.
P2. Environnement & Scénarios de Simulation	3-4	Mettre en place un environnement de test reproductible et créer des scénarios d'attaque validés.	**2.1. Installation et configuration de LocalStack/Docker** et de la base de données Graphe (Neo4j ou équivalent). **2.2. Création de 3 scénarios d'attaque CTI** (Simple, Complexe, Multi-principal). **2.3. Validation manuelle de l'escalade de privilèges** pour chaque scénario.	Environnement de Lab (Docker Compose). 3 Fichiers de configuration IAM (JSON/YAML) pour les scénarios.
P3. Développement du Moteur de Cartographie	5-6	Développer le composant d'ingestion des données IAM et de construction du Graphe.	**3.1. Script de collecte des configurations IAM** (via API LocalStack). **3.2. Script de conversion des données brutes en nœuds/arêtes du Graphe**. **3.3.**	Code source du module `iam_collector.py`. Base de données Graphe peuplée avec les scénarios P2.

			Implémentation de la persistance du Graphe.	
P4. Moteur de Détection & Validation	7-8	Implémenter l'algorithme de recherche de chemins d'attaque et valider l'outil.	**4.1. Algorithme de recherche de chemins** (ex: Cypher) pour identifier les chaînes CTI. **4.2. Tests unitaires et de validation** sur les scénarios P2. *Optionnel : Fonctionnalité de "scoring" des chemins*.	Code source du module `attack_path_finder.py` . Rapport de validation (Validation.md).
P5. Documentation Technique & Stratégies de Défense	9-10	Rédiger la documentation technique de l'outil et les recommandations de mitigation.	**5.1. Rédaction du README.md** (Installation, Utilisation, Exemples). **5.2. Rédaction du chapitre "Défense et Mitigation"** (Principes de moindre privilège, Boundary Policies). *Optionnel : Préparation de la présentation (slides) pour la reconnaissance Industrie*.	Documentation Open-Source (README, CONTRIBUTING) . Document de Recommandations Sécurité (Mitigation.md).
P6. Publication & Finalisation	11-12	Préparer les livrables académiques et la publication open-source.	**6.1. Rédaction de l'article/mémoire de recherche** (Introduction,	Article/Mémoire de Recherche (Draft final). Dépôt GitHub public.

			Modèle, Implémentation, Résultats, Conclusion). **6.2. Nettoyage et mise en licence du code source** (MIT/Apache 2.0). **6.3. Préparation du dépôt GitHub final**.	

Séparation des Livrables : MVP vs. Optionnel

Pour garantir la réussite du projet dans le délai imparti, une distinction claire est faite entre le Produit Minimum Viable (MVP) et les fonctionnalités optionnelles.

Catégorie	Indispensable (MVP)	Optionnel (Nice-to-have)
Portée Technique	Formalisation CTI pour **un seul fournisseur Cloud** (AWS via LocalStack). Détection des chemins d'escalade de privilèges.	Extension de la formalisation/simulation à GCP ou Azure. Interface de visualisation graphique du graphe. Fonctionnalité de *scoring* des chemins d'attaque.
Livrables	CTI-Model.md, Docker Compose, Scénarios IAM, `iam_collector.py`, `attack_path_finder.py`, Validation.md, README.md, Mitigation.md, Draft Académique.	Présentation Industrie (Slides), Soumission à un workshop/conférence.
Objectif	Prouver la faisabilité de la formalisation et de la détection.	Augmenter l'impact et la portée de la recherche.

Points Critiques et Gestion des Risques

L'identification et la mitigation des risques sont essentielles pour la gestion de projet. Trois risques majeurs ont été identifiés :

Risque	Description	Impact Potentiel	Stratégie de Mitigation
R1. Dérive de la Modélisation	La formalisation théorique (P1) s'avère trop complexe ou ne correspond pas aux spécificités réelles de l'IAM Cloud.	Retard dans le développement du moteur de cartographie (P3) et remise en cause de la validité académique.	**Priorité au MVP AWS :** Se concentrer sur les primitives IAM d'AWS (via LocalStack) pour le MVP. Valider le modèle avec des exemples concrets et publics dès P1.
R2. Problèmes d'Environnement	Difficultés à simuler fidèlement l'IAM Cloud avec LocalStack ou à intégrer la base de données Graphe (Neo4j/équivalent).	Blocage complet des phases de développement (P3, P4).	**Validation Précoce :** Tester l'intégration LocalStack/Graphe dès la semaine 3. Utiliser des images Docker éprouvées. Prévoir une alternative simple (ex: stockage des relations dans un format JSON/CSV) si l'intégration Graphe échoue.
R3. Faux Positifs/Négatifs	Le moteur de détection (P4) génère trop de bruit ou manque des chemins d'attaque réels.	Remise en cause de la validité de l'outil et du papier académique.	**Critères Stricts :** Définir des critères de "chemin d'attaque valide" très stricts basés sur la formalisation P1. Multiplier les tests de validation sur les scénarios P2.

Conclusion et Roadmap Résumée

Ce plan de projet de 12 semaines est une feuille de route rigoureuse qui mène de la **modélisation théorique** (P1) à la **publication open-source et académique** (P6). L'ordre de

priorité est optimal, car chaque phase technique (P3, P4) dépend directement de la phase théorique (P1) et de la phase de préparation de l'environnement (P2).

La réussite du projet repose sur le respect strict du périmètre MVP (focalisation sur AWS simulé) et sur la gestion proactive des risques, notamment la validation précoce de l'environnement de simulation.

Semaines	Phase	Objectif Principal	Livrable Clé
1-4	Théorie & Setup (P1-P2)	Formalisation de l'attaque et mise en place de l'environnement de test.	CTI-Model.md, Docker Compose, Scénarios IAM
5-8	Développement (P3-P4)	Construction de l'outil de cartographie et du moteur de détection.	`iam_collector.py`, `attack_path_finder.py`, Validation.md
9-12	Documentation & Publication (P5-P6)	Rédaction des livrables finaux pour l'industrie et l'académie.	README.md, Mitigation.md, Article Draft, Dépôt GitHub Public