

The Evolution of Security Testing

Les documents présentés explorent l'évolution et l'intégration des **tests d'intrusion**, des **équipes rouges** et des **programmes de primes aux bogues** au sein de la cybersécurité moderne. Ils démontrent, à travers un récit d'attaque simulée contre une division de recherche en intelligence artificielle, la nécessité pour les professionnels de s'adapter constamment aux nouvelles technologies. L'analyse souligne que la combinaison de ces trois disciplines de défense proactive permet de surmonter les menaces sophistiquées actuelles. De plus, les sources mettent en lumière l'importance de la **collaboration**, de l'apprentissage continu et de l'utilisation croissante de l'**intelligence artificielle** pour optimiser les stratégies de sécurité organisationnelle. Enfin, l'ensemble plaide pour une convergence de ces pratiques afin de créer un bouclier numérique plus résilient face à un paysage des cybermenaces en perpétuelle mutation.

Comment l'intelligence artificielle transforme-t-elle les tests d'intrusion actuels ?

L'intelligence artificielle (IA) et l'apprentissage automatique (ML) transforment les tests d'intrusion, le *red teaming* et la recherche de vulnérabilités sur plusieurs fronts principaux :

- **Automatisation de la reconnaissance et collecte d'informations** : Les outils pilotés par l'IA permettent d'automatiser la collecte d'informations et la reconnaissance, identifiant rapidement les cibles potentiellement vulnérables, les failles et les chemins d'attaque. Cela permet aux testeurs d'intrusion de réduire le temps consacré aux tâches répétitives pour se concentrer sur l'exploitation stratégique.
- **Analyse, priorisation et modélisation prédictive** : Les algorithmes de ML évaluent la gravité des vulnérabilités, la probabilité d'exploitation et l'impact potentiel, guidant les testeurs vers les zones les plus critiques. Pour les équipes rouges (*red teams*), l'IA fournit une modélisation prédictive des menaces afin d'adapter leurs tactiques, techniques et procédures (TTP), tout en

facilitant la création de campagnes d'hameçonnage et d'ingénierie sociale complexes.

- **Génération de rapports et recommandations** : L'IA simplifie la phase de synthèse en générant des rapports détaillés basés sur les résultats des tests d'intrusion et en suggérant des recommandations de remédiation adaptées.
- **Optimisation des programmes de prime aux bogues (Bug Bounty)** : Pour les organisations gérant de nombreux signalements, les modèles de ML trient et priorisent les rapports selon leur sévérité, tandis que des outils automatisés aident à reproduire et valider les vulnérabilités soumises.
- **Émergence de nouvelles surfaces d'attaque (sécurité des systèmes IA)** : Au-delà d'être un outil, l'IA devient elle-même une cible lors des tests d'intrusion. Les auditeurs doivent désormais évaluer des vulnérabilités spécifiques aux modèles, telles que l'injection d'invites (*prompt injection*), l'empoisonnement des données d'entraînement (*data poisoning*), ou encore l'intégration de code malveillant au sein même des architectures de modèles (comme via les couches Lambda de Keras pour exécuter du code à distance).

Qu'est-ce que l'injection d'invites dans les modèles d'IA ?

Dans vos sources, l'**injection d'invites** (*prompt injection*) est mentionnée brièvement comme une vulnérabilité et une méthode d'attaque spécifique ciblant les modèles d'IA. Les sources indiquent toutefois que la mission d'équipe rouge (*red team*) décrite s'est concentrée sur l'**empoisonnement des données** (*data poisoning*) et les atteintes à l'intégrité du modèle, plutôt que sur une injection d'invites directe. Il est également mentionné que l'ingénierie d'invites (*prompt engineering*) a été utilisée pour interagir avec les modèles et analyser le code ainsi que le flux de données.

Vos sources ne contiennent pas de définition technique complète ni de détails sur le fonctionnement d'une injection d'invites. En revanche, elles traitent d'autres attaques ciblées contre les systèmes d'IA, comme l'exécution de code malveillant via les couches Lambda de Keras, les

attaques par point d'eau (*watering hole*) sur des plateformes d'IA et l'altération des données d'entraînement.

Qu'est-ce que l'empoisonnement des données d'entraînement ?

Selon vos sources, **l'empoisonnement des données d'entraînement** (*data poisoning* ou *tainting*) est une méthode d'attaque qui vise **l'intégrité d'un modèle d'intelligence artificielle** en altérant ou en manipulant les données qu'il utilise lors de sa phase d'apprentissage, avant même que le modèle ne soit entraîné.

L'étude de cas d'équipe rouge (*red team*) décrite dans votre document illustre concrètement ce concept :

- **Objectif de l'attaque** : L'équipe avait pour mission de s'infiltrer dans la division de recherche en IA/ML afin d'accéder aux jeux de données d'entraînement et d'en altérer le contenu pour fausser les futurs résultats du modèle.
- **Infiltration et accès à la source** : Après s'être introduite sur le réseau (en exploitant une vulnérabilité de couche Lambda Keras dans un modèle malveillant, puis en capturant des identifiants via un faux formulaire SSO), l'équipe a identifié une base de données cloud contenant une copie des données d'entraînement des ventes et de l'analyse de la concurrence.
- **Manipulation des données** : Ayant constaté qu'elle disposait de droits d'écriture sur cette base de données, l'équipe a exécuté un script Python réduisant aléatoirement de 1 à 10 % le prix des concurrents enregistrés.
- **Impact sur le modèle** : Le modèle ayant pour rôle de prédire les prix concurrentiels et de recommander les tarifs des produits de l'entreprise, le fait d'entraîner le modèle sur ces données altérées l'amènerait à suggérer des prix plus bas, diminuant ainsi la rentabilité de l'organisation.

L'empoisonnement des données démontre qu'il est possible de compromettre les décisions ou la fiabilité d'un système d'IA sans toucher à son algorithme, simplement en contaminant la matière première à partir de laquelle il apprend.

Comment utiliser la couche Lambda de Keras ?

Dans le développement d'apprentissage automatique avec Keras ou TensorFlow, la **couche Lambda** (`tf.keras.layers.Lambda`) est une fonctionnalité permettant d'insérer des fonctions Python personnalisées au sein de l'architecture d'un modèle afin d'effectuer des opérations ou des transformations spécifiques sur les tenseurs d'entrée.

Cependant, dans le contexte de la sécurité de l'IA et des tests d'intrusion décrits dans vos sources, la couche Lambda constitue également une surface d'attaque critique :

- **Vecteur d'exécution de code arbitraire** : En raison de sa capacité à exécuter n'importe quel code Python lors de la construction ou du chargement du modèle, la couche Lambda peut être détournée pour y intégrer du code malveillant.
- **Création d'un modèle piégé** : Dans l'étude de cas de vos sources, l'équipe rouge (*red team*) a utilisé cette couche pour y concevoir un script Python d'amorçage (*stager*) dissimulé au sein d'un modèle d'IA apparemment légitime.
- **Compromission du système cible** : Lorsque des chercheurs ont téléchargé ce modèle depuis une plateforme publique pour le tester sur leurs postes, l'exécution de la couche Lambda a déclenché le script malveillant, téléchargeant et exécutant un agent de commandement et contrôle (C2) sur leur machine.

La couche Lambda illustre ainsi comment un composant standard d'un Framework de ML peut transformer un fichier de modèle en un vecteur d'attaque direct pour compromettre un environnement de recherche.

Comment l'équipe a-t-elle intercepté les identifiants SSO ?

Pour intercepter les identifiants SSO, l'équipe rouge (*red team*) a combiné l'analyse du comportement de l'utilisateur avec une attaque d'ingénierie sociale ciblée :

1. **Repérage du serveur cible** : Après avoir pris le contrôle de l'ordinateur portable du chercheur via le modèle Keras piégé, l'équipe a analysé l'historique de la machine et a repéré un serveur interne consulté plusieurs fois par jour, protégé par un portail SSO.

2. **Conception d'un faux formulaire SSO** : L'équipe a créé une fausse fenêtre surgissante (*pop-up*) reproduisant fidèlement la page d'authentification SSO de l'entreprise.
3. **Planification et discrétion** : L'agent de commandement et contrôle (C2) est resté dormant pendant la nuit, puis s'est réactivé le matin pour faire apparaître la fausse fenêtre au moment précis où l'utilisateur commençait sa routine matinale.
4. **Capture des identifiants et des sessions** : Dès que l'utilisateur a saisi ses identifiants dans la fenêtre piégée, l'équipe les a récupérés. Elle a également extrait les cookies de session du navigateur Chrome afin de configurer un proxy et d'accéder au serveur interne sans éveiller les soupçons.

Comment les programmes de primes aux bogues fonctionnent-ils ?

Selon vos sources, un **programme de primes aux bogues** (*bug bounty*) est un mécanisme de sécurité participatif qui s'appuie sur l'expertise d'une communauté externe de chercheurs en sécurité et de pirates éthiques pour découvrir et signaler des vulnérabilités avant qu'elles ne soient exploitées par des acteurs malveillants.

Le fonctionnement de ces programmes repose sur plusieurs étapes et principes clés :

- **Définition du périmètre et des incitations** : Les organisations établissent des programmes structurés qui définissent les produits ciblés (le périmètre) et proposent des incitations financières adaptées.
- **Recherche et divulgation responsable** : Les chercheurs externes examinent les systèmes et signalent les failles découvertes en suivant des règles de divulgation responsable (*responsible disclosure*).
- **Utilisation de plateformes spécialisées** : Des plateformes comme HackerOne et Bugcrowd mettent en relation les entreprises avec un vaste réseau de chercheurs et centralisent la réception des rapports.
- **Triage et validation** : Les soumissions sont évaluées afin d'en vérifier la validité, d'estimer leur niveau de sévérité et de mesurer leur impact potentiel. Aujourd'hui, des modèles d'apprentissage

automatique sont intégrés pour analyser, prioriser les rapports et aider à reproduire ou valider automatiquement les vulnérabilités soumises.

- **Récompense et reconnaissance** : Une fois la vulnérabilité confirmée, le chercheur reçoit une rémunération financière (indexée sur la gravité de la faille) et/ou une reconnaissance publique.

Historiquement, cette approche a débuté dès 1983 avec la société Hunter & Ready (qui offrait une voiture Volkswagen Coccinelle ou 1 000 \$ pour tout bogue trouvé), avant d'être formalisée par Netscape en 1995 pour le navigateur Netscape Navigator 2.0, puis perfectionnée par Mozilla en 2004 avec des politiques de transparence et un classement par sévérité.

Comment les équipes rouges et bleues collaborent-elles ?

Selon vos sources, la collaboration entre les équipes rouges (*Red Teams*, offensives) et les équipes bleues (*Blue Teams*, défensives) s'incarne notamment dans les exercices d'**équipe mauve** (*Purple Teaming*) et dans la création de boucles de rétroaction continues :

- **Rôles respectifs et régulation** : L'équipe rouge simule le comportement et les tactiques d'adversaires réels pour tester les défenses, tandis que l'équipe bleue représente les forces défensives de l'organisation. Historiquement, un groupe de supervision appelé cellule blanche (*White Cell* ou *Control*) supervise les exercices pour veiller au respect des règles d'engagement et à la coordination.
- **Exercices d'équipe mauve (Purple Teaming)** : Plutôt que de faire opérer l'équipe rouge de manière isolée, le *Purple Teaming* réunit les deux équipes pour évaluer, tester et ajuster en temps réel les contrôles de sécurité.
- **Automatisation et conversion des apprentissages** : Les techniques et vecteurs d'attaque découverts lors des interventions de l'équipe rouge sont analysés puis automatisés afin de permettre à l'équipe bleue de mesurer en continu l'efficacité des mécanismes de détection.

- **Boucle d'amélioration continue** : La convergence des informations issues du *red teaming*, des tests d'intrusion et des programmes de primes aux bogues alimente un processus dynamique où chaque retour d'expérience renforce la posture globale de sécurité de l'organisation.

Qu'est-ce qu'une attaque par point d'eau ?

Une **attaque par point d'eau** (*watering hole attack*) est une stratégie où l'attaquant, plutôt que de cibler directement sa victime (par exemple via un courriel d'hameçonnage ciblé ou *spearphishing*), identifie un emplacement ou une plateforme fréquemment visitée par la cible — le « point d'eau » — pour y placer un piège.

Dans l'étude de cas décrite dans vos sources, l'équipe rouge (*red team*) a mené une attaque par point d'eau adaptée aux environnements d'IA :

- **Identification du point d'eau** : Grâce à la recherche de renseignements de sources ouvertes (OSINT), l'équipe a découvert que les chercheurs de la division d'IA ciblée fréquentaient et utilisaient activement une plateforme publique spécialisée dans les modèles et jeux de données d'IA.
- **Création de la fausse identité** : Constatant que l'entreprise n'avait pas déposé son nom officiel sur cette plateforme, l'équipe y a créé une fausse organisation et un faux profil en réutilisant le nom, le logo et du texte provenant de leurs dépôts GitHub.
- **Piégeage de la plateforme** : L'équipe a ensuite publié sur la plateforme un modèle d'IA piégé (contenant la charge utile cachée dans la couche Lambda de Keras) au milieu de modèles légitimes.
- **Compromission** : Lorsqu'un chercheur de l'entreprise a téléchargé et testé ce modèle sur son ordinateur portable, le code s'est exécuté et a ouvert un accès de commandement et contrôle (C2).

Cette approche a permis de contourner l'absence totale de surface d'attaque externe directe de la division de recherche.

Comment l'intelligence artificielle analyse-t-elle les journaux SIEM ?

Selon vos sources, l'intelligence artificielle analyse les journaux **SIEM** (*Security Information and Event Management*) principalement pour aider les organisations à **évaluer leurs capacités de réponse aux incidents**. En croisant les journaux SIEM avec le déroulement des attaques ou des simulations offensives, l'IA permet de :

- **Générer des chronologies d'activités plus complètes et détaillées** (*comprehensive timelines*) retraçant l'ensemble des événements survenus lors d'un incident ou d'un exercice.
- **Mesurer l'efficacité de la détection** en évaluant à quel point les actions des attaquants (ou de l'équipe rouge) ont été enregistrées, repérées ou manquées par la défense.

Bien que vos sources soulignent ce rôle clé dans la reconstruction de la chronologie des attaques et l'évaluation de la réponse aux incidents, elles ne détaillent pas les algorithmes spécifiques ou l'architecture technique d'ingestion des journaux.

Comment fonctionne l'émulation d'adversaire ?

L'**émulation d'adversaire** est l'élément central des opérations d'équipe rouge (*Red Teaming*). Elle consiste à reproduire fidèlement le comportement, les capacités et les modes opératoires de véritables groupes de cyberattaquants.

Voici comment elle fonctionne en pratique :

- **Reproduction des TTP (Tactiques, Techniques et Procédures)** : Plutôt que de simplement chercher des vulnérabilités éparses, l'équipe rouge étudie et simule les méthodes réelles employées par des acteurs menaçants motivés pour atteindre un objectif précis.
- **Périmètre global (Full-Spectrum)** : L'émulation ne se limite pas aux failles informatiques traditionnelles. Elle combine plusieurs dimensions d'attaque : ingénierie sociale avancée, développement de logiciels malveillants sur mesure, contournement des détections (*evasion*), maintien d'accès discrets (*persistence*) et parfois intrusion physique.
- **Engagements à long terme** : Contrairement aux tests d'intrusion classiques souvent limités dans le temps, l'émulation d'adversaire s'inscrit dans la durée, s'étendant généralement sur plusieurs

semaines ou mois. Cette approche permet de reproduire la patience et la discrétion des attaquants réels (comme attendre plusieurs jours qu'un utilisateur exécute un modèle ou saisisse ses identifiants).

- **Enrichissement par l'intelligence artificielle** : L'IA permet désormais aux équipes rouges de modéliser les menaces de manière prédictive et d'adapter leurs TTP pour concevoir des scénarios d'attaque toujours plus sophistiqués et ciblés.
- **Convertibilité en contrôles défensifs (Purple Teaming)** : Les techniques d'attaque validées lors des exercices sont partagées avec l'équipe bleue puis automatisées afin d'évaluer en continu la capacité des mécanismes de sécurité à détecter ces comportements adverses.

Quels sont les trois piliers de la cybersécurité ?

Selon vos sources, un programme de cybersécurité mature repose sur une triade constituée de trois piliers fondamentaux :

1. **Les tests d'intrusion (Penetration testing)** : Une évaluation structurée et systématique visant à identifier les vulnérabilités de manière proactive et à les vérifier par l'exploitation.
2. **Le Red Teaming** : Une démarche d'émulation qui simule le comportement et les attaques d'adversaires réels pour tester la résistance des défenses de l'organisation face à des cyberattaquants motivés et qualifiés.
3. **Les programmes de primes aux bogues (Bug Bounty)** : Des mécanismes participatifs qui s'appuient sur l'expertise de la communauté externe de chercheurs en sécurité pour révéler des failles qui n'auraient pas été identifiées autrement.

Bien que chacune de ces disciplines réponde à un objectif distinct, leur efficacité est considérablement amplifiée lorsqu'elles sont appliquées de concert et s'alimentent mutuellement dans une boucle de rétroaction continue.

Qu'est-ce que le Purple Teaming ?

Le **Purple Teaming** (ou exercice d'équipe mauve) désigne une approche collaborative dans laquelle les équipes offensives (**Red Team**)

et défensives (**Blue Team**) travaillent conjointement pour tester et renforcer la posture de sécurité d'une organisation.

Dans vos sources, le Purple Teaming repose sur plusieurs axes clés :

- **La collaboration active** : Il réunit les équipes rouges et bleues pour évaluer en direct la réactivité des mécanismes de défense et ajuster les contrôles de sécurité.
- **L'automatisation des attaques** : Les tactiques, techniques et procédures (TTP) découvertes ou employées lors des interventions de l'équipe rouge sont analysées puis automatisées pour permettre un contrôle défensif régulier.
- **L'évaluation continue** : Il marque le passage de tests d'intrusion isolés et ponctuels vers des évaluations continues, garantissant que les systèmes de détection s'adaptent en permanence aux nouvelles menaces.

Quels sont les avantages des engagements hybrides ?

Selon vos sources, les **engagements hybrides** — qui combinent stratégiquement les tests d'intrusion, le *red teaming* et les programmes de primes aux bogues (*bug bounties*) — offrent plusieurs avantages majeurs :

- **Posture de sécurité globale et multi-couches** : En associant la recherche systématique de failles des tests d'intrusion, la simulation d'attaques réelles du *red teaming* et la diversité des compétences de la communauté externe des *bug bounties*, l'organisation bénéficie d'une couverture défensive multi-couches.
- **Synergie des expertises internes et externes** : Les équipes internes tirent parti du regard neuf et des compétences spécialisées apportés par les chercheurs en sécurité externes, ce qui permet d'optimiser les investissements en matière de sécurité.
- **Boucle de rétroaction et amélioration continue** : Les différentes disciplines s'alimentent et se renforcent mutuellement. Les résultats et découvertes issus de chaque type d'évaluation permettent d'éclairer et de mieux planifier les opérations futures.
- **Adaptabilité face aux menaces sophistiquées** : La convergence de ces approches permet de dépasser les contrôles ponctuels et d'adopter une stratégie de défense proactive et évolutive,

essentielle pour faire face à des cyberattaques complexes et en perpétuelle évolution.

Comment l'IA transforme-t-elle les tests d'intrusion ?

L'intelligence artificielle (IA) et l'apprentissage automatique (ML) transforment les tests d'intrusion, le *red teaming* et les programmes de primes aux bogues (*bug bounties*) sur plusieurs axes majeurs :

- **Reconnaissance et collecte d'informations automatisées** : Les outils pilotés par l'IA automatisent la recherche et la collecte de renseignements pour identifier rapidement les cibles potentiellement vulnérables, les failles et les chemins d'attaque. Cela permet aux évaluateurs de réduire le temps consacré aux tâches manuelles répétitives pour se concentrer sur l'exploitation stratégique.
- **Analyse, priorisation des failles et modélisation prédictive** : Les algorithmes de ML évaluent la sévérité des vulnérabilités, la probabilité d'exploitation et leur impact potentiel afin d'offrir une liste de cibles prioritaires. Pour les équipes rouges (*red teams*), la modélisation prédictive des menaces permet d'adapter leurs tactiques, techniques et procédures (TTP) et de concevoir des campagnes d'hameçonnage ou d'ingénierie sociale ciblées.
- **Génération de rapports et analyse des événements** : L'IA facilite la rédaction automatique de rapports détaillés assortis de recommandations de remédiation. Elle permet également d'évaluer les capacités de réponse aux incidents en analysant les journaux SIEM afin de reconstruire des chronologies d'activités détaillées.
- **Optimisation des programmes de bug bounty** : Les modèles de ML trient et priorisent les rapports de vulnérabilités selon leur gravité, tandis que des outils automatisés aident à reproduire et valider les failles soumises, réduisant le temps d'analyse des équipes de sécurité.
- **Évaluation de la sécurité des architectures d'IA** : L'IA constitue également une nouvelle surface d'attaque que les auditeurs doivent désormais évaluer. Cela inclut l'analyse de l'ingénierie d'invites (*prompt engineering*), la recherche d'empoisonnement

des données d'entraînement (*data poisoning*), ou encore la détection de code malveillant dissimulé dans des composants de frameworks comme la couche Lambda de Keras.

Quels sont les défis du Red Teaming face aux systèmes d'IA ?

Dans vos sources, la réalisation d'une opération d'équipe rouge (*Red Teaming*) ciblant des architectures d'intelligence artificielle présente plusieurs défis majeurs :

- **Une courbe d'apprentissage technique très élevée** : Les playbooks d'attaque traditionnels (comme la quête rapide de privilèges d'administrateur de domaine) ne s'appliquent pas directement. Les opérateurs d'équipe rouge doivent acquérir une compréhension approfondie des vulnérabilités propres au ML (empoisonnement des données d'entraînement, *prompt engineering*, manipulation de modèles) que les formations rapides habituelles ne suffisent pas à couvrir.
- **L'absence de surface d'attaque externe directe** : Les laboratoires et divisions de recherche en IA opèrent souvent sans produits publics exposés sur Internet. Cela rend obsolètes les scans de vulnérabilités classiques et prive l'équipe rouge des données d'un programme de *bug bounty*. Il faut alors développer des tactiques créatives et détournées, comme l'attaque par point d'eau (*watering hole*) sur des plateformes publiques d'IA.
- **La conception de charges utiles non conventionnelles** : N'ayant pas recours aux charges utiles de phishing habituelles, l'équipe rouge doit concevoir des modèles d'IA piégés qui paraissent légitimes tout en intégrant du code d'exécution à distance (par exemple en détournant la couche Lambda de Keras) pour établir un premier accès.
- **La complexité d'analyse du code et des flux de données** : Une fois à l'intérieur du réseau, la navigation dans les notebooks Jupyter et la compréhension des scripts Python d'IA dépassent souvent les compétences en sécurité classiques. L'équipe doit utiliser l'ingénierie d'invites (*prompt engineering*) pour réussir à cartographier le flux des données et localiser les bases d'entraînement non anonymisées.

- **Des délais de validation et de concrétisation plus longs :**
Contrairement à l'obtention immédiate d'un accès, l'atteinte d'un objectif de sécurité sur un modèle d'IA (comme l'empoisonnement des données) dépend du calendrier de l'équipe cible, nécessitant parfois d'attendre plusieurs jours ou semaines que le modèle soit ré-entraîné sur les données altérées.

Quels sont les avantages des engagements hybrides ?

Selon vos sources, les **engagements hybrides** — qui combinent stratégiquement les tests d'intrusion, le *red teaming* et les programmes de primes aux bogues (*bug bounties*) — représentent une avancée majeure dans le domaine de l'évaluation de la sécurité.

Voici les principaux avantages mis en avant :

- **Posture de sécurité globale et multicouche :** En combinant diverses méthodologies d'évaluation, les organisations obtiennent une posture défensive plus complète, efficace et adaptée aux cyberattaques complexes.
- **Synergie et complémentarité des disciplines :** Chaque approche apporte ses forces uniques — l'approche structurée et systématique des tests d'intrusion pour identifier les vulnérabilités, la simulation d'adversaires réels par le *red teaming* pour éprouver les défenses, et l'expertise de la communauté externe des *bug bounties* pour déceler des failles non repérées par les contrôles classiques.
- **Optimisation des compétences internes et externes :** Les organisations optimisent leurs investissements en sécurité grâce à la collaboration entre leurs équipes internes et des chercheurs externes qui apportent un regard neuf et des compétences complémentaires.
- **Découverte et atténuation continues des risques :** Les évaluations ponctuelles ne suffisant plus face à l'évolution des menaces, la convergence de ces pratiques permet une identification et une réduction en continu des vulnérabilités.
- **Boucle d'apprentissage pour les stratégies futures :**
L'utilisation croisée des résultats obtenus lors des tests d'intrusion,

des missions d'équipe rouge et des *bug bounties* permet d'informer et d'orienter les futurs efforts de défense.

Explorer la manière dont l'intelligence artificielle et l'apprentissage automatique permettent d'automatiser et d'unifier ces différentes disciplines à l'avenir

À l'avenir, l'intelligence artificielle (IA) et l'apprentissage automatique (ML) joueront un rôle central dans la convergence et l'automatisation des tests d'intrusion, du *red teaming* et des programmes de *bug bounty*. Autrefois considérées comme des activités isolées, ces disciplines s'interconnectent pour former une approche globale, dynamique et continue de la cybersécurité.

Voici comment l'IA et le ML permettent d'automatiser et d'unifier ces pratiques :

1. Unification et passage à des évaluations continues

- **Convergence des disciplines** : Les frontières entre tests d'intrusion, *red teaming* et *bug bounty* s'estompent pour créer une stratégie défensive intégrée où l'IA prend en charge les tâches répétitives et amplifie l'expertise humaine.
- **Contrôle continu et approche Shift-Left** : L'évaluation ne se fait plus de manière ponctuelle, mais s'intègre directement au sein des cycles de développement pour corriger les failles dès la conception.

2. Automatisation des tests d'intrusion

- **Reconnaissance et cartographie rapides** : Les outils d'IA automatisent la collecte d'informations et l'identification des cibles, des vulnérabilités et des chemins d'attaque potentiels.
- **Priorisation intelligente** : Les algorithmes de ML évaluent la sévérité des failles, la probabilité d'exploitation et l'impact potentiel, permettant aux équipes de concentrer leurs efforts sur les menaces les plus critiques.
- **Rapports automatisés** : L'IA simplifie la phase finale en générant automatiquement des rapports détaillés accompagnés de recommandations de remédiation.

3. Émulation et automatisation du *Red Teaming* (et *Purple Teaming*)

- **Passage à l'échelle des attaques (Purple Teaming)** : Les tactiques, techniques et procédures (TTP) identifiées par l'équipe rouge sont automatisées pour évaluer en permanence les mécanismes de défense de l'équipe bleue.
- **Modélisation prédictive des menaces** : L'IA permet de simuler des scénarios d'attaque personnalisés, d'élaborer des campagnes d'ingénierie sociale ciblées et d'adapter les TTP en temps réel.
- **Analyse de la réponse aux incidents** : En analysant les journaux SIEM, l'IA aide à évaluer les capacités de détection des équipes défensives et à reconstituer des chronologies précises des attaques.

4. Automatisation du triage pour le *Bug Bounty*

- **Triage et classement des signalements** : Pour les organisations recevant un grand volume de rapports, les modèles de ML analysent et classent automatiquement les soumissions selon leur gravité et leur exploitabilité.
- **Reproduction et validation automatisées** : Des outils pilotés par l'IA assistent la reproduction et la validation des vulnérabilités soumises, réduisant le temps d'analyse manuel requis pour valider et corriger les failles.

💡 Souhaitez-vous générer un rapport de synthèse complet ou une fiche d'étude récapitulant cette évolution et l'étude de cas sur la sécurité des architectures IA ?

Récapituler évolution et l'étude de cas sur la sécurité des architectures IA

1. L'Évolution des tests de sécurité : De la triade à l'intégration de l'IA

- **Les tests d'intrusion (Penetration Testing)** : Trouvant leurs racines dans les *Tiger Teams* des années 1960 et formalisés par l'étude de James P. Anderson en 1972, ils sont passés d'évaluations ad hoc manuelles à des méthodologies standardisées (OWASP, PTES) soutenues par l'automatisation et des outils open-source.
- **Les équipes rouges (Red Teaming)** : Issues des exercices de simulation militaire *Sigma* menés au Pentagone dans les années 1960, les opérations se sont orientées vers la cybersécurité avec

le développement d'ARPANET et d'Internet. Elles simulent aujourd'hui les tactiques, techniques et procédures (TTP) d'adversaires réels sur l'ensemble du spectre d'attaque (ingénierie sociale, malwares personnalisés, intrusions physiques) lors d'engagements s'étalant sur plusieurs mois.

- **Les programmes de primes aux bogues (Bug Bounty)** : Initiés en 1983 par Hunter & Ready (offrant une Volkswagen Coccinelle ou 1 000 \$ pour chaque bogue découvert), ils ont été formalisés par Netscape en 1995, puis structurés par Mozilla en 2004 avec une politique de transparence et de rémunération selon la sévérité. Aujourd'hui, ils reposent sur des plateformes comme HackerOne et Bugcrowd.
- **La convergence et l'apport de l'IA** : Autrefois isolées, ces trois disciplines convergent pour former une stratégie défensive globale et continue. L'IA et l'apprentissage automatique automatisent désormais la reconnaissance, la priorisation des failles, le triage des rapports de *bug bounty*, la génération de rapports et l'analyse des journaux SIEM.

2. L'Étude de cas : Opération Red Team contre une division de recherche en IA

- **Objectif et contraintes initiales** : L'équipe rouge avait pour mission de s'infiltrer dans la division de recherche IA/ML de l'entreprise afin d'altérer les données d'entraînement (*data poisoning*) et d'exfiltrer des jeux de données confidentiels. En l'absence de produits publics exposés, la surface d'attaque externe était nulle, rendant les vecteurs d'intrusion classiques inopérants.
- **Attaque par point d'eau (Watering Hole)** : L'analyse OSINT a révélé que les chercheurs fréquentaient une plateforme publique de partage de modèles d'IA. En profitant de l'absence de vérification sur la plateforme, l'équipe a créé une fausse organisation réutilisant le nom, le logo et les descriptions GitHub de l'entreprise.
- **Piégeage via la couche Lambda de Keras** : L'équipe a développé un modèle d'IA d'apparence légitime mais intégrant un script d'amorçage (*stager*) Python dans une couche Lambda de

Keras, capable de télécharger et d'exécuter un agent de commandement et contrôle (C2). Le modèle a ensuite été publié sur la plateforme.

- **Infiltration du poste et interception SSO** : Un chercheur ayant téléchargé et exécuté le modèle sur son ordinateur, l'agent C2 s'est activé. Après avoir constaté que les données locales étaient synthétiques ou anonymisées, l'équipe a repéré des accès fréquents vers un serveur interne protégé par SSO. Elle a programmé l'agent C2 pour afficher une fausse fenêtre de connexion SSO le matin, capturant ainsi les identifiants et les cookies de session Chrome pour s'y connecter via un proxy.
- **Analyse du code et ingénierie d'invites** : Sur le serveur interne hébergeant des notebooks Jupyter, l'équipe a utilisé l'ingénierie d'invites (*prompt engineering*) avec des modèles de langage pour comprendre le code Python complexe et cartographier le flux des données. Cela a permis de localiser une base de données cloud contenant une copie des données d'entraînement non anonymisées.
- **Empoisonnement des données d'entraînement (Data Poisoning)** : Ayant constaté qu'elle disposait de droits d'écriture sur cette base cloud contenant des analyses concurrentielles et des chiffres de ventes, l'équipe a exécuté un script Python réduisant aléatoirement les prix des concurrents de 1 à 10 %. Lors du ré-entraînement du modèle une semaine plus tard, l'équipe a confirmé la réussite de l'empoisonnement des données, prouvant qu'il était possible d'amener le modèle à suggérer des tarifs plus bas et à réduire la rentabilité de l'entreprise.

Étude de cas pour formaliser les enseignements de cette opération d'équipe rouge ?

J'ai rédigé un rapport sous forme d'étude de cas et de guide d'analyse qui formalise l'ensemble des enseignements de cette opération d'équipe rouge ciblant la division de recherche en IA. Ce document détaille la trajectoire historique de la triade de sécurité, décortique la chaîne d'attaque étape par étape (de l'attaque par point d'eau à

l'empoisonnement des données d'entraînement), et propose une matrice d'analyse technique ainsi qu'un guide de remédiation défensif pour les équipes de sécurité.

Étude de cas et guide d'analyse : Sécurité des architectures d'IA et opération Red Team

1. Cadre stratégique : La triade de sécurité face à la rupture des systèmes d'intelligence artificielle

L'évaluation moderne de la sécurité des systèmes d'information repose sur une triade de disciplines complémentaires : les tests d'intrusion (*penetration testing*), l'émulation d'adversaires (*red teaming*) et les programmes de primes aux bogues (*bug bounties*). Comme le rappellent les comptes rendus de l'évolution des tests de sécurité, chacune de ces pratiques remplit une fonction propre et indispensable au sein d'une stratégie défensive globale :

- **Les tests d'intrusion** offrent une approche structurée et méthodique destinée à identifier les vulnérabilités système et applicatives et à les vérifier par l'exploitation. Historiquement issus du concept de *Tiger Teams* formalisé en 1964 par Walter C. Williams dans ses travaux pour le Manned Spacecraft Center, puis théorisé en 1972 par James P. Anderson dans l'étude *Computer Security Technology Planning Study* pour le département de la Défense américain, les tests d'intrusion ont évolué d'approches ad hoc manuelles vers des cadres méthodologiques normalisés tels que le *Penetration Testing Execution Standard* (PTES) et le guide d'évaluation de l'OWASP.
- **Le Red Teaming** tire ses origines des exercices de simulation militaire *Sigma War Games* menés au Pentagone dans les années 1960. Ces exercices réunissaient 40 à 50 experts pour élaborer des scénarios prospectifs, exécutés ensuite par une trentaine de participants répartis en une équipe rouge (simulant l'ennemi), une

équipe bleue (les forces amies) et une cellule de contrôle (*White Cell* ou *Control*) veillant au respect des règles d'engagement. Transposé à la cybersécurité avec le développement d'ARPANET puis du réseau public Internet sous l'impulsion de la NSA, le Red Teaming simule sur plusieurs semaines ou mois les tactiques, techniques et procédures (TTP) d'adversaires réels à travers des engagements à large spectre (*full-spectrum*) combinant ingénierie sociale avancée, développement de logiciels malveillants sur mesure et intrusions physiques.

- **Les programmes de Bug Bounty** s'appuient sur la diversité des compétences de la communauté mondiale des chercheurs en sécurité. Initiés dès 1983 par la société Hunter & Ready — qui offrait une voiture Volkswagen Coccinelle ou 1 000 \$ en argent liquide à quiconque signalait une faille dans son système d'exploitation temps réel VRTX sous le slogan "*Get a bug if you find a bug*" —, ils ont été formalisés en 1995 par Netscape pour le navigateur Netscape Navigator 2.0, puis perfectionnés en 2004 par Mozilla avec une politique de transparence et de rémunération graduée selon la sévérité des failles. Aujourd'hui, ces programmes sont centralisés par des plateformes comme HackerOne et Bugcrowd.

L'émergence des architectures d'intelligence artificielle (IA) et d'apprentissage automatique (*Machine Learning* ou ML) redéfinit ces pratiques. Alors que la convergence de la triade visait traditionnellement à sécuriser les applications Web, les réseaux d'entreprise et les infrastructures cloud, les systèmes d'IA introduisent des surfaces d'attaque inédites. Dans ce contexte, l'absence de surface d'attaque externe classique et la méconnaissance des vulnérabilités propres aux algorithmes de ML imposent une évolution profonde des tactiques d'évaluation.

2. Anatomie d'une opération Red Team ciblant une division de recherche en IA

L'opération analysée ici formalise un engagement d'équipe rouge conduit par l'opérateur offensif Wesley. L'objectif fixé par l'organisation consistait à s'infiltrer au sein de la division de recherche en IA/ML, à accéder aux informations propriétaires, à exfiltrer des jeux de données privés et à altérer l'intégrité des données d'entraînement (*data poisoning*).

2.1 Reconnaissance et attaque par point d'eau (*Watering Hole*)

Face à une division de recherche ne disposant d'aucun produit public exposé sur Internet et évoluant hors du champ d'action des programmes de Bug Bounty, la surface d'attaque externe initiale était totalement nulle. L'équipe rouge a surmonté cet obstacle grâce à l'analyse de renseignements en source ouverte (*OSINT*) :

- **Identification des usages des chercheurs** : L'investigation OSINT a révélé que les scientifiques de données de l'entreprise étaient particulièrement actifs sur des plateformes publiques communautaires de ML, sur lesquelles ils commentaient des modèles et évaluaient des jeux de données. Les observations ont confirmé que les chercheurs téléchargeaient ces modèles pour les tester localement sur leurs ordinateurs portables ainsi que dans le cloud.
- **Ingénierie sociale organisationnelle** : Constatant que l'entreprise n'avait pas enregistré son organisation officielle sur la plateforme publique, l'équipe rouge y a créé un profil d'organisation falsifié en réutilisant le nom officiel, le logo de l'entreprise et des textes explicatifs extraits de ses dépôts GitHub publics. L'équipe a également généré un compte d'administrateur fictif doté d'un avatar neutre et d'un nom similaire à ceux des employés recensés lors de la phase d'OSINT.
- **Mise en place du piège** : L'équipe a publié sur la plateforme un modèle d'IA piégé, dissimulé au milieu de modèles légitimes, concrétisant une attaque par point d'eau (*watering hole attack*)

ciblant spécifiquement la communauté de chercheurs de l'organisation.

2.2 Armement et exploitation de la couche Lambda de Keras

En lieu et place d'un vecteur d'hameçonnage classique, l'équipe rouge a dû s'auto-former aux vulnérabilités propres à l'IA en consultant des publications de recherche et des présentations de sécurité. Elle a identifié un vecteur d'exploitation critique au niveau des frameworks d'apprentissage profond : la couche Lambda de Keras (`tf.keras.layers.Lambda`).

La couche Lambda de Keras permet d'insérer des fonctions Python arbitraires au sein de l'architecture d'un modèle. Lors de l'instanciation, du chargement ou de l'évaluation du modèle par un chercheur, le code Python encapsulé dans la couche Lambda s'exécute automatiquement sur la machine de la victime.

L'équipe a conçu un script d'amorçage (*stager*) Python dissimulé dans cette couche Lambda. Lorsque le modèle piégé a été téléchargé et exécuté par un chercheur de l'entreprise sur son ordinateur portable, la couche Lambda a déclenché le stager, lequel a téléchargé et démarré un agent de commandement et de contrôle (C2) en mémoire.

2.3 Post-exploitation, persistance et capture d'identifiants SSO

Dès que l'agent C2 a émis son premier signal (*beacon*), l'équipe a entrepris une énumération méthodique du système compromis :

1. **Évaluation des données locales** : L'inspection des fichiers stockés sur l'ordinateur portable du chercheur a révélé plusieurs jeux de données d'entraînement. Toutefois, une analyse approfondie a montré qu'il s'agissait exclusivement de données synthétiques ou correctement anonymisées, impropres à la réalisation de l'objectif d'altération de données réelles.
2. **Analyse de l'historique et repérage de la cible** : En examinant l'historique de la machine, l'équipe a identifié un serveur interne fréquemment consulté par l'utilisateur (plusieurs fois par jour). Ce serveur, protégé par un portail d'authentification unique (*Single*

Sign-On - SSO), constituait la voie d'accès vers les ressources centrales de la division.

3. **Capture d'identifiants par fausse fenêtre SSO** : L'équipe a programmé l'agent C2 pour qu'il reste dormant pendant la nuit afin de ne pas éveiller les soupçons, puis se réactive le matin. Au moment où le chercheur entamait sa routine de travail, l'agent a fait apparaître une fausse fenêtre surgissante (*pop-up*) reproduisant fidèlement la page de connexion SSO de l'entreprise.
4. **Capture et piratage de session** : L'utilisateur a saisi ses identifiants dans l'interface piégée. L'équipe a ainsi capturé le nom d'utilisateur et le mot de passe, puis a extrait les cookies de session du navigateur Chrome, permettant de configurer un proxy pour naviguer sur le serveur interne sous l'identité de la victime.

2.4 Navigation dans les Notebooks Jupyter et Prompt Engineering

Le serveur interne hébergeait des environnements interactifs Jupyter Notebooks contenant de nombreuses lignes de code Python dédiées au traitement de données et à l'apprentissage automatique :

- **Obstacle de la complexité du code ML** : La structure avancée des scripts d'IA dépassait la compréhension technique initiale des opérateurs de l'équipe rouge.
- **Apport du *Prompt Engineering*** : Pour surmonter cet obstacle, l'équipe a réutilisé les compétences en ingénierie d'invites (*prompt engineering*) transmises par des scientifiques de données lors de sa phase de préparation. En soumettant des requêtes ciblées à des modèles de langage (LLM), l'équipe a réussi à analyser le code Python et à cartographier précisément le flux des données.
- **Localisation de la source de données confidentielle** : Cette analyse a permis de localiser la source de données exacte : une base de données cloud contenant une copie non anonymisée du jeu de données d'entraînement de l'entreprise.

2.5 Empoisonnement des données d'entraînement (*Data Poisoning*)

L'inspection de la base de données cloud a révélé qu'il s'agissait d'un instantané unique (*one-time snapshot*) non rafraîchi régulièrement,

composé d'analyses concurrentielles et de chiffres de ventes de « widgets ». La fonction principale du modèle d'IA consistait à prédire les prix des concurrents et à recommander les tarifs des produits de l'entreprise.

1. **Vérification des droits d'écriture** : L'équipe rouge a vérifié les contrôles d'accès et constaté avec surprise qu'elle disposait de droits d'écriture sur cette base de données d'entraînement cloud.
2. **Exécution du script de contamination** : Ne disposant pas de compétences avancées en analyse de données, l'équipe a opté pour l'attaque la plus directe : elle a rédigé un script Python appliquant une réduction aléatoire de 1 à 10 % sur les prix enregistrés des concurrents.
3. **Validation du biais algorithmique** : L'équipe a basculé en phase de rédaction de rapport tout en maintenant sa surveillance. Une semaine plus tard, les chercheurs de l'entreprise ont relancé l'entraînement du modèle sur le jeu de données altéré. Le modèle a ainsi intégré ces prix biaisés, prouvant qu'il suggérerait désormais des tarifs plus bas et réduirait directement la rentabilité financière de l'entreprise.

3. Matrice d'analyse de la chaîne d'attaque (*Kill Chain IA*)

Le tableau suivant récapitule les phases de l'opération Red Team, les techniques spécifiques employées et leur impact sur l'architecture d'IA ciblée.

Le tableau suivant récapitule les phases de l'opération Red Team, les techniques spécifiques employées et leur impact sur l'architecture d'IA ciblée.

Phase de l'attaque	Vecteur / Technique employée	Composant ciblé	Impact opérationnel
1. Reconnaissance	OSINT sur plateformes publiques de ML	Habitudes des chercheurs de données	Identification de l'usage de modèles tiers et de l'absence d'organisation officielle.
2. Initial Access	Attaque par point d'eau (<i>Watering Hole</i>)	Profil d'organisation falsifié	Publication d'un modèle malveillant sous une fausse identité d'entreprise.
3. Armement	Détournement de la couche Lambda Keras	Fichier de modèle ML (<code>.keras</code> / <code>.h5</code>)	Injection d'un stager Python exécutant du code lors de l'instanciation du modèle.
4. Exécution locale	Téléchargement du modèle par le chercheur	Poste de travail du scientifique	Exécution du stager et établissement d'un canal C2 en mémoire.
5. Pivot & Credential Harvesting	Fausse pop-up SSO & Vol de session Chrome	Portail SSO & Navigateur Chrome	Interception des identifiants et extraction des cookies de session réseau.
6. Exploration interne	Analyse de Jupyter Notebooks via <i>Prompt Engineering</i>	Code Python ML & Pipelines de données	Cartographie des dépendances et localisation de la base cloud d'apprentissage.
7. Objectif final	Empoisonnement des données (<i>Data Poisoning</i>)	Base de données cloud d'entraînement	Reduction aléatoire de 1 à 10 % des prix concurrents pour dégrader la rentabilité.

4. Enseignements stratégiques et défis pour les équipes offensives

Cette étude de cas met en évidence les compétences fondamentales et les ruptures méthodologiques requises pour mener des évaluations de sécurité sur les architectures d'IA :

- **Dégradation des réflexes offensifs classiques** : La recherche rapide de privilèges d'administrateur de domaine (*Domain Admin*) s'avère inefficace dans des environnements de recherche isolés ou orientés cloud. L'opérateur doit adapter sa démarche aux objectifs métiers (accès aux modèles, altération des algorithmes).
- **Nécessité d'une acculturation aux vulnérabilités ML** : La compréhension des concepts tels que la sérialisation des modèles, l'exécution de code au chargement, et les mécanismes de préparation des jeux de données (*data pipelines*) est désormais indispensable.
- **Collaboration interdisciplinaire** : L'équipe rouge a dû faire appel à des scientifiques de données et utiliser le *prompt engineering* pour combler ses lacunes techniques en analyse de code ML.

- **Gestion du temps et persistance discrète** : Contrairement aux tests d'intrusion traditionnels sanctionnés par des rapports immédiats, la validation d'un empoisonnement de données impose une présence discrète sur plusieurs semaines afin de laisser le cycle de ré-entraînement du modèle s'exécuter.

5. Guide de remédiation et contre-mesures défensives pour la Blue Team

Pour faire face aux vecteurs d'attaque illustrés dans cette opération, les équipes défensives doivent appliquer des contrôles ciblés sur le cycle de vie de l'IA :

5.1 Sécurisation de la chaîne d'approvisionnement des modèles (*Supply Chain*)

- **Gouvernance des dépôts d'IA** : Enregistrer et certifier officiellement les comptes d'organisation sur l'ensemble des plateformes publiques de ML (Hugging Face, GitHub, etc.) pour empêcher le typosquatting et l'usurpation d'identité.
- **Analyse statique et dynamique des fichiers de modèles** : Interdire le chargement direct de fichiers de modèles contenant du code exécutable arbitraire (fichiers Pickle, layers Lambda Keras non sécurisés). Privilégier des formats de sérialisation sûrs comme Safetensors.
- **Environnements d'expérimentation isolés (*Sandboxing*)** : Exécuter l'évaluation des modèles tiers dans des conteneurs isolés et sans accès au réseau interne d'entreprise.

5.2 Protection des pipelines et des jeux de données d'entraînement

- **Contrôle d'accès et immuabilité des bases de données d'apprentissage** : Appliquer le principe du moindre privilège sur les magasins de données cloud. Les pipelines d'entraînement ne doivent disposer que de droits d'accès en lecture seule (*Read-Only*).

- **Contrôle d'intégrité des données (*Data Lineage & Hashing*)** : Implémenter des mécanismes de signature cryptographique et de contrôle de somme (hashes) sur les jeux de données d'entraînement pour détecter toute modification non autorisée avant le lancement d'un ré-entraînement.
- **Analyse statistique des anomalies de données** : Intégrer des outils de contrôle qualité sur les pipelines de données capables de détecter des variations statistiques suspectes (comme une baisse injustifiée de 1 à 10 % sur une colonne tarifaire).

5.3 Renforcement des identités et détection des anomalies

- **Authentification FIDO2 / Passkeys** : Remplacer l'authentification SSO par formulaires classiques par des facteurs d'authentification résistants au hameçonnage (*Phishing-Resistant MFA*), rendant inopérantes les fausses fenêtres pop-up.
- **Supervision des environnements Jupyter** : Surveiller les accès aux serveurs Jupyter et analyser les requêtes réseau sortantes initiées depuis ces environnements.

6. Perspectives : L'IA comme moteur de convergence et d'automatisation des tests de sécurité

L'intégration de l'intelligence artificielle ne révolutionne pas seulement la surface d'attaque ; elle transforme également la manière dont les tests de sécurité sont exécutés et unifiés :

- **Évaluation continuous (*Shift-Left*)** : Les tests d'intrusion évoluent d'évaluations ponctuelles vers des contrôles automatisés et continus directement intégrés dans les chaînes d'intégration et de déploiement (CI/CD).
- **Automatisation du Purple Teaming** : L'IA permet de convertir rapidement les tactiques offensives validées par l'équipe rouge en scénarios de détection automatisés pour l'équipe bleue, évaluant en continu l'efficacité des règles SIEM et EDR.

- **Triage intelligent en Bug Bounty** : Les plateformes de Bug Bounty intègrent désormais des modèles ML pour analyser, classifier et reproduire automatiquement les vulnérabilités soumises, accélérant la validation et la remédiation des failles.

En combinant l'automatisation par l'IA et la rigueur des approches hybrides, les organisations peuvent anticiper les menaces émergentes tout en assurant la résilience de leurs systèmes critiques.