Management et Gouvernance de la Sécurité

ICS - Informatique et Cybersécurité
M. LANGLOY et Mme COUDER

VIII - Audit de sécurité

Afin de pouvoir mettre en œuvre les méthodes abordées précédemment (ERM, référentiels, mesures, etc.), il est nécessaire d'avoir une **idée très précise de l'état du SI** au moment de la prise de décision. Ce besoin d'un « relevé photographique » est orienté, il dépend de la stratégie suivie, et n'est donc presque jamais disponible en tant que tel. **Établir cet instantané selon le but précis recherché**, est ce que l'on nomme **mener un audit**.

L'audit de sécurité s'appuie sur **toutes sortes de données préalables** : la documentation de l'élément audité, les pratiques de déploiement, de développement, de production, les extraits de configuration, l'accès aux interfaces d'administration, etc. La collecte de ces **informations multiples et hétérogènes** est nécessaire à la production d'un résultat efficace et performant.
Par ailleurs, la plupart des entreprises possède des tableaux de bord permettant de piloter l'état de leur SI. Ces outils de **monitoring** sont des éléments extrêmement intéressants dans la mise en œuvre **d'audits longs** ou **continus**.

1. Généralités

L'audit de sécurité a pour but de **vérifier l'état de la sécurité** de tout ou partie d'un système, à **un instant T**, en comparaison à un **état de référence**.
Il répertorie ses **points forts** et surtout ses **points faibles** (vulnérabilités).
Généralement, l'auditeur dresse également une série de **recommandations** (mesures) dans le **rapport d'audit**.

Un audit est **déclenché par un besoin** : l'**analyse d'un risque** et/ou pour vérifier la **conformité à un attendu** :

- une norme (telle qu'ISO 27 002), un référentiel (ex : COBIT), un guide de bonnes pratiques (comme les Global Technology Audit Guides, le CISA, le guide d'hygiène de l'ANSSI, etc.) ;
- un texte de loi ;
- un règlement interne à l'entreprise ;
- la Politique de Sécurité du SI (PSSI) ;
- une base documentaire du SI ;
- etc.

Un audit peut ainsi être effectué pour répondre à des **problématiques diverses** :

- repérer précisément les actifs de l'entreprise (logiciels, bases de données, matériels informatiques,…) ;
- recenser les menaces à un instant donné ;
- mesurer les impacts des vulnérabilités et consolider le SI ;
- définir des parades pour renforcer la résilience informatique et permettre à l'entreprise de fonctionner en cas d'attaque (chiffrement de données, sauvegardes, plan de secours,…) ;
- etc.

Il est fréquent d'auditer les éléments suivants :

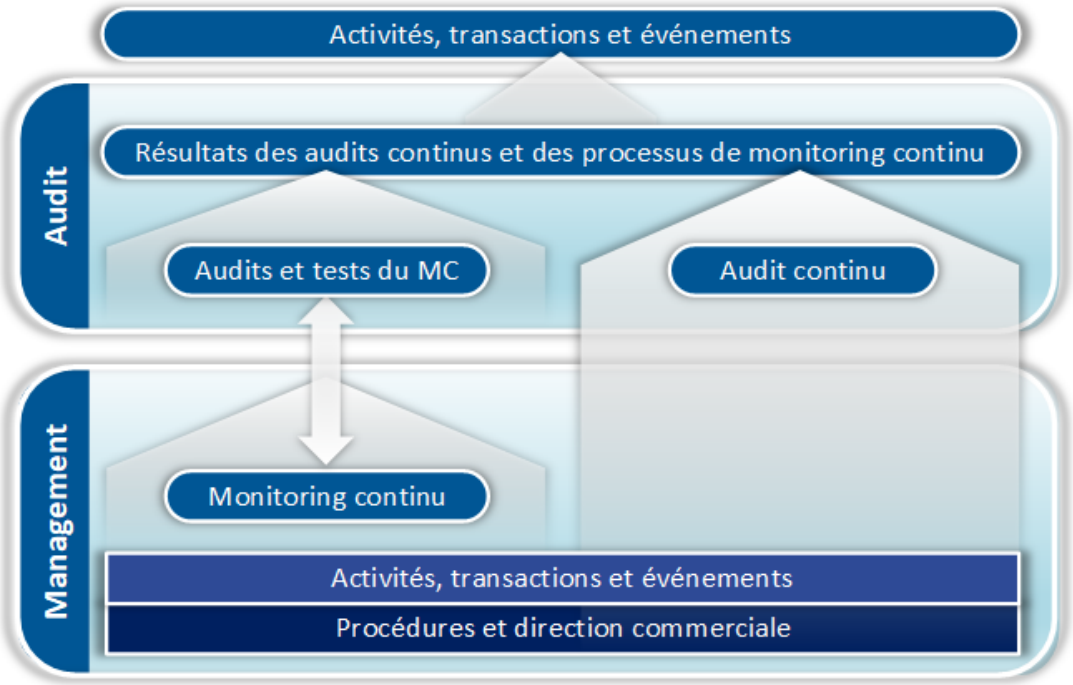
Actif audité	Exemples
L'organisation, les processus et les contrôles	- les comptes et les privilèges - les modifications du SI (arrivée, départ, changement) - les droits sur les ressources sensibles - la documentation technique - la politique de gestion des authentifications par défaut - les usages d'administration sur les postes - la politique de mise à jour du SI - la gestion des logs sensibles - les plans d'action (sauvegardes, PCA, PRA, PAS,...) - les procédures de gestion d'incidents de sécurité - la charte informatique
Les équipements physiques	- la politique de sécurité des locaux, dont la gestion des visiteurs - les habilitations d'accès physique du personnel, plus particulièrement aux salles accueillant du matériel stratégique tel que des serveurs - la protection contre les supports amovibles - la sécurisation physique des terminaux nomades - la politique de sécurité particulière aux utilisateurs en déplacement
L'architecture	- le filtrage appliqué aux points d'entrées / sorties du SI - la protection des crédencials (« login + mot de passe + domaine ») - le processus d'authentification forte - le niveau minimal de sécurité exigé - la gestion des sécurités locales, telles que le pare-feu - le chiffrement des données sensibles - la protection de la messagerie - la sécurisation des connexions locales / distantes ; filaire / Wifi - les services visibles depuis Internet - le niveau de sécurité des protocoles utilisés - la sécurisation de l'administration - la segmentation et le cloisonnement des réseaux, notamment du réseau d'administration - l'isolation de l'administration des Systèmes
Les partenaires	- la sécurisation des interconnexions avec les partenaires - l'anticipation des fins de contrats (notamment de maintenance) - les adhérences logicielles (dépendances de fonctionnement d'un logiciel par rapport à un autre)

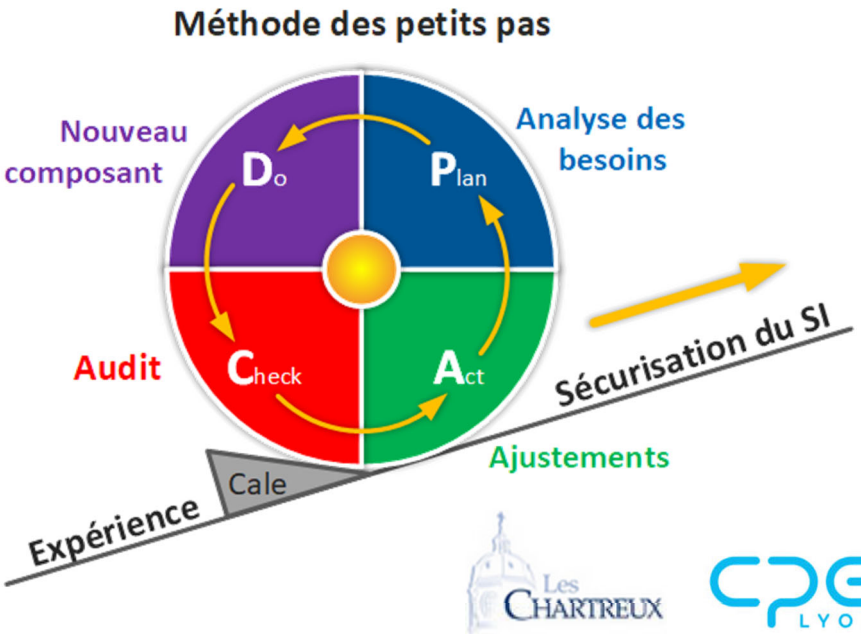
De façon générale, un audit a pour **objectifs** de :

Inventorier et identifier les **actifs critiques** du SI	Classifier et **prioriser** les vulnérabilités	Lier les **mesures correctives** appropriées

Attention : l'audit ne fait pas l'analyse de risques ! Certes, il permet de **trouver les vulnérabilités**, mais **pas de déterminer si celles-ci sont tolérables**. L'auditeur est souvent un prestataire extérieur et c'est bien le client (audité) qui décide selon sa stratégie de sécurité de **suivre ou non ses recommandations**.

L'audit est plus généralement utilisé en appui du responsable de la sécurité informatique, de **façon ponctuelle** pour répondre à un besoin spécifique, ou dans une **démarche de surveillance continue** des risques :





L'audit est également préconisé dans la démarche d'amélioration continue représentée par la **Roue de Deming** (ou **PDCA**), en **vérification (C)** après la mise en œuvre de tout **nouveau composant de sécurité (D)**.

2. Mise en œuvre d'un audit

Typologie

Nous l'avons évoqué, un audit dépend de **l'environnement** et de **l'objectif** dans lesquels il est mené. Pour qu'il soit pertinent et adapté, il est d'abord nécessaire de bien comprendre ses caractéristiques :

Audit passif	Audit actif
Reconnaissance passive avec **prise d'empreintes** Uniquement basé sur des **observations** **Pas d'interaction** avec l'environnement ➢ L'objectif est d'obtenir et de lister des **données factuelles** lié à l'objectif. Ces techniques s'apparentent à de l'**OSINT**.	**Recherche** de la compromission ➢ L'objectif est de **faire réagir le SI** par divers procédés pour constater son niveau de vulnérabilité en rapport avec l'objectif.

Audit automatique	Audit manuel
Basé sur des **applicatifs ou scripts** permettant de vérifier les **problématiques connues** **Indépendant de l'environnement** de l'entreprise **Rapide** mais standardisé	Basé sur **l'analyse et une interprétation** de l'environnement de l'entreprise Nécessite des **ressources intellectuelles avancées** Plus pointu et **pertinent**, mais assez long

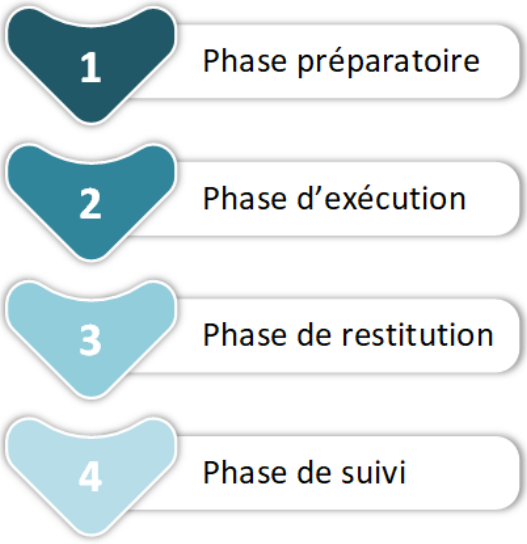
Audit interne	Audit externe
Étude **depuis l'intérieur de l'entreprise** dans les conditions environnementales liées à l'objectif	Évaluation du **potentiel de vulnérabilité** face aux problématiques liées à une **personne étrangère** à l'entreprise. ➢ L'objectif est d'identifier la **perméabilité du SI** (pénétration pour l'entrant / fuite pour le sortant)

Audit technique	Audit global
Problématiques d'**installation**, de **configuration** et du **fonctionnement technique** du SI	**Appréciation des usages** et de la compréhension de ceux-ci Problématiques liées à la **logique**

Bien sûr, un audit est constitué d'une **pluralité de caractéristiques**.

Démarche

Quels que soient l'audit et ses caractéristiques, on applique généralement une démarche en phases qui s'enchainent de la préparation aux résultats obtenus.



Phase préparatoire

Organisation d'une réunion de cadrage entre **l'auditeur** et **l'organisme client audité** (éventuellement son commanditaire s'il s'agit d'un audit externe).
Cette rencontre doit permettre de définir les objectifs de l'audit, son périmètre, sa portée, ses critères, sa planification, etc.
Préparer revient à **élaborer une stratégie générale** de réalisation de l'audit.

Les objectifs de la planification sont les suivants :

- fixer les rôles et les responsabilités des parties prenantes à l'exercice, et mettre en place une **stratégie de coordination d'audit** ;
- prendre connaissance de la mission et des processus de l'organisme à auditer ;
- s'informer à propos des **réglementations** auxquelles est soumise l'organisation ou l'unité sur laquelle porte l'audit ;
- définir **l'attendu auquel se réfèrera l'auditeur** pour formuler une opinion quant aux écarts de pratiques constatés au sein de l'entreprise - cette phase devra préciser des **éléments probants, pertinents et suffisants** pour évaluer la conformité à cet attendu ;
- acquérir une **bonne connaissance de l'environnement** visé par l'audit, tant informatique que dans ses aspects de gouvernance si cela est pertinent (vision, stratégie, objectifs, etc.) ;
- mieux comprendre la **structure organisationnelle** de l'entreprise, les **compétences** et l'**expérience** du personnel à qui sont confiées les tâches liées à la sécurité ;
- identifier les **contraintes** qui pèsent sur l'audit (ex : prévoir la non-disponibilité de certaines personnes, évaluer les ressources humaines et matérielles nécessaires, etc.) ;
- préparer le **programme d'audit**, c'est-à-dire la synthèse des étapes clés de la réalisation de l'audit, les objectifs, les points de contrôle, les tests et les ressources nécessaires.

Une évaluation des risques peut être réalisée durant la phase préparatoire, afin de prioriser les efforts déployés pour assurer la gestion du personnel nécessaire.
Dans le cas où l'entreprise a déjà fait une évaluation des risques, l'auditeur pourrait bien sûr s'en servir pendant son audit.

Phase d'exécution

Réalisation des **travaux de terrain**, assez fréquemment avec l'aide d'**outils de collecte** d'informations.
Le contenu de cette phase dépend fortement des caractéristiques et de la portée de l'audit.

Phase de restitution

Remise du rapport d'audit comportant les **recommandations** de l'auditeur, lors d'une réunion de clôture à laquelle assistent généralement la Direction et le DSI côté client.
Il est important que les constats, méthodes et conclusions de l'audit soient bien comprises à cette étape par toutes les parties prenantes du projet.

Phase de suivi

Pour terminer, il appartient à l'organisation auditée de **mettre en application** les recommandations jugées pertinentes et d'assurer le **suivi de leur mise en œuvre effective**.
Ce suivi s'effectue généralement par un plan d'action listant les tâches affectées aux personnes responsables de la mise en place des recommandations, et leurs délais.

Exemple

Pour un **audit d'intrusion**, AlgoSecure propose cette méthodologie :

Préalablement aux tentatives d'intrusion, nous **définissons avec vous le périmètre à étudier** : votre site institutionnel, des applications web spécifiques, un ensemble de périphériques, une plage d'adresses réseau, des bâtiments physiques, etc.
Nous définissons également **le niveau d'informations dont nos pentesters bénéficient** au commencement des tests : boite noire (aucune information), boite grise (accès à des comptes d'utilisateurs) ou boite blanche (accès aux comptes d'administrateurs et plus).

Le jour J, nous organisons une **réunion de lancement** avec vos interlocuteurs techniques, puis, après une ultime confirmation, nous débutons les **tests techniques**.

En boite noire, nos tests d'intrusion débutent typiquement par une phase de **reconnaissance passive** où nous nous renseignons sur la cible sans établir de connexion avec elle. S'en suit une phase de **prise d'empreintes** durant laquelle nous listons les services et les applications exposées.
Pour chaque service découvert, nous **cherchons** s'il existe des **vulnérabilités connues**, des défauts de configuration, des failles, etc. et nous identifions les points d'entrée utilisateur que nous pourrions exploiter pour compromettre le périmètre.

En boite grise, où nous avons généralement un ou plusieurs comptes à notre disposition, nous vérifions de surcroit que les données et fonctionnalités sont **correctement cloisonnées** (contrôle d'accès), et qu'un **déplacement latéral** ou qu'une **élévation de privilèges** n'est pas possible.

En boite blanche, nous vérifions également le **contrôle d'accès du compte d'administration** et que les permissions accordées à celui-ci ne soient pas suffisamment larges pour s'échapper de l'application et **prendre le contrôle du serveur sous-jacent**.
Si nous avons du **code source à disposition**, nous en effectuons une rapide **étude** orientée sécurité, afin de chercher davantage de vulnérabilités.

Enfin, nous vous récapitulons les principaux constats de l'audit durant un **point de synthèse**, avant de terminer par la **rédaction des livrables**.

Méthodes

Pour arriver à dresser une liste la plus exhaustive possible des vulnérabilités d'un système, différentes pratiques existent et sont traditionnellement mises en œuvre conjointement.

Entretien

L'entretien est une **approche globale**, particulièrement **nécessaire** dans le cadre d'un **audit de sécurité organisationnel**, qui consiste à rencontrer les personnes pour bien comprendre leur rôle dans le contrôle de sécurité.

Rappelons que l'humain est la faille principale dans une entreprise ; **toutes les personnes** ayant un **rôle à jouer dans la sécurité** du SI doivent donc être interrogées :

- le Directeur des Systèmes d'Information (DSI) ;
- les Responsables de la Sécurité des Systèmes d'Information (RSSI) ;
- les administrateurs ;
- les utilisateurs du SI, qu'ils aient un rôle dans la production de l'entreprise, dans la gestion ou la simple utilisation des moyens informatiques ;
- tout autre personne ayant un lien avec la sécurité.

En plus des failles de sécurité, un audit révèle très souvent un **besoin d'accompagnement** et de **montée en compétences** en cybersécurité des équipes. Il est important de formuler les **questions avec tact**. En effet, interroger des personnes à propos de leur travail peut faire qu'elles se sentent jugées et les résultats peuvent en être faussés. **La diplomatie est** donc une compétence **essentielle pour la pratique des audits**.

Relevé de configurations

Autre composante nécessaire à un audit de sécurité : **l'analyse de l'architecture et des composants du SI**. On relève ici dans le détail leurs caractéristiques que l'on compare à des systèmes sécurisés et aux failles et faiblesses les plus courantes.

Tout peut être inspecté, de l'architecture réseau du SI aux systèmes et aux applications.
Par exemple sur un serveur, les points suivants sont généralement analysés :

- le chargeur de démarrage ;
- les mécanismes d'authentification (robustesse des mots de passe, utilisation d'authentification forte, etc.) et les comptes d'accès ;
- le système de fichiers (permissions, utilisation de chiffrement etc.) ;
- les services ;
- la journalisation ;
- etc.

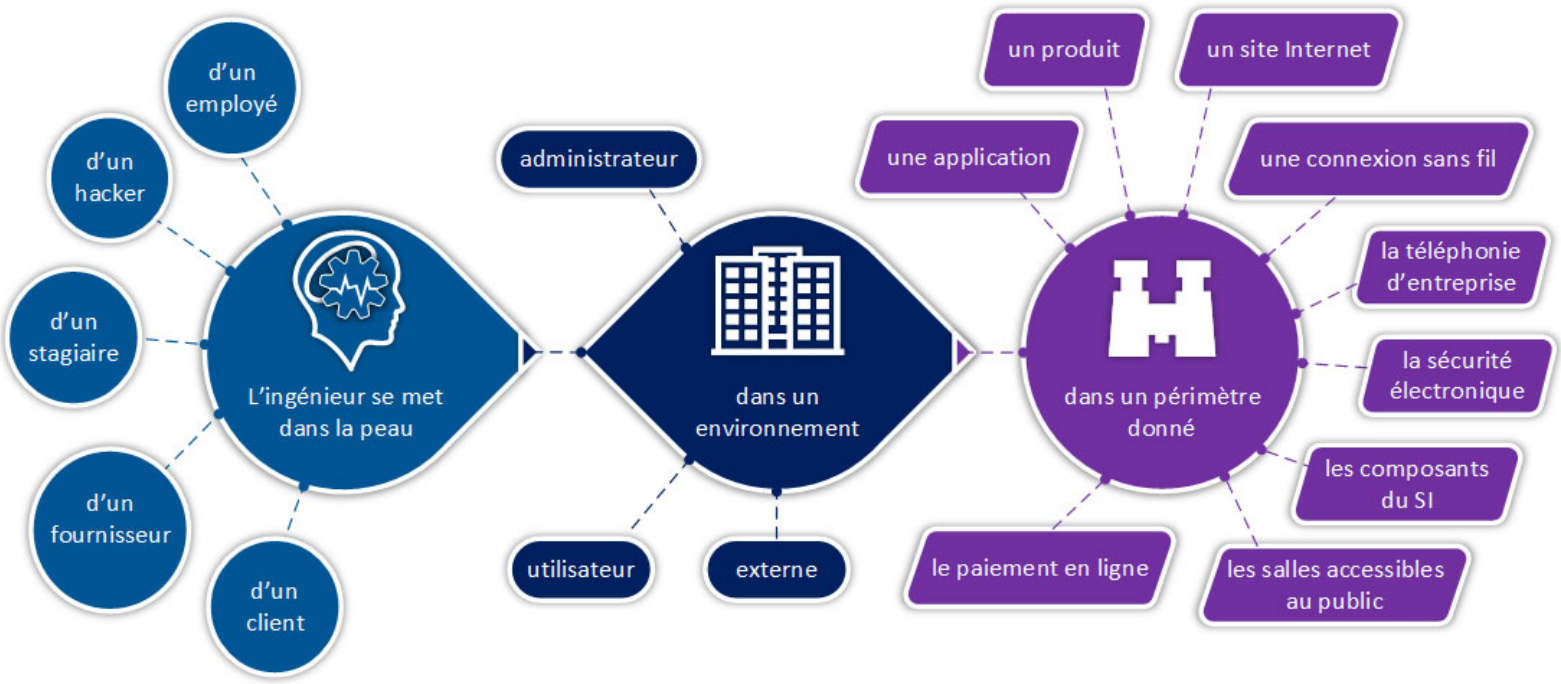
Tests d'intrusion (Pentest)

Les tests d'intrusion sont une pratique **d'audit technique**, qui propose une approche par la **théorie des systèmes** selon laquelle tout phénomène doit être considéré comme un système (ou peut être conceptualisé selon une logique de système), c'est-à-dire comme un **ensemble complexe d'interactions**. C'est le contraire d'une vue analytique ou cartésienne.

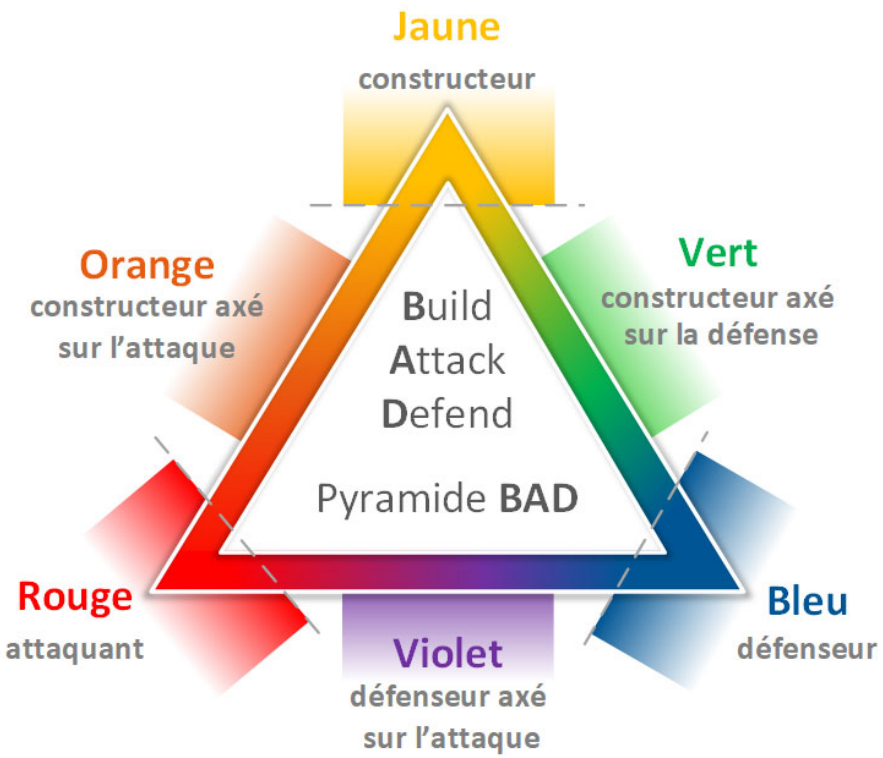
Cette méthode, adossée à une **logique réductionniste** consiste à découper un problème en petites parties, puis à **analyser chacune d'elles individuellement**, sans se préoccuper du fonctionnement global de l'ensemble. C'est grâce à cela que les petits blocages méthodiques, passés jusque-là presque inaperçus, apparaissent clairement et peuvent être exploités.

La méthode consiste généralement à analyser l'infrastructure du SI, afin de **simuler une attaque** par un utilisateur mal intentionné ou un logiciel malveillant.
Lors d'un test d'intrusion, le **pentester** adopte la position d'un **hacker potentiel** dans le but de trouver des vulnérabilités exploitables et d'élaborer un plan d'actions permettant d'améliorer la sécurité du SI, sans attendre qu'un vrai pirate informatique compromette les infrastructures internes de l'entreprise.

Au préalable de tests d'intrusion, il faut définir leur **rôle du pentester**, les **conditions environnementales** et les **objectifs** des tests, selon une approche scénaristique :



Pour identifier le **rôle**, les **enjeux** et les **objectifs** de chaque partie prenante dans cette approche, il est fréquent de représenter **chaque équipe** par une **couleur** sur le triptyque BAD (construire / attaque / défendre) :



Équipe rouge - attaquants

Anciennement « **équipe tigre** », elle a pour mission de **tester l'efficacité d'un programme de sécurité** en reproduisant les outils et les techniques des attaquants probables, de la manière la plus **réaliste** possible.
La pratique est similaire (mais pas identique) à des tests de pénétration, et vise la poursuite d'objectifs précis, habituellement exécutés en campagne.

Équipe bleue - défenseurs

Équipe de sécurité interne qui **se défend contre les vrais attaquants** et l'équipe rouge.
L'équipe bleue se distingue des équipes de sécurité classiques par sa mentalité de **vigilance constante** contre les attaques.
Ses meilleurs membres emploient « **l'empathie adversariale** », c'est-à-dire la faculté de **penser comme l'ennemi**, acquise grâce à leur **expérience** des attaques.

Équipe jaune - constructeurs

Généralement les développeurs de logiciels et les architectes responsables de la conception des réseaux ou de la mise en œuvre d'autres actifs.

Équipe violette - défenseurs ayant appris des attaquants

Équipe chargée de **maximiser l'efficacité des équipes rouge et bleu**.
Elle le fait en combinant les tactiques et les contrôles défensifs de l'équipe bleue avec les menaces et les vulnérabilités trouvées par l'équipe rouge, dans un seul récit qui maximise les deux (ex : informer le Bleu des attaques de Rouge pour qu'elle se défende au mieux ; informer Rouge des zones protégées pour qu'il attaque ailleurs).
Idéalement, le Violet ne devrait pas être une équipe, mais une **dynamique permanente entre le Rouge et le Bleu**.

Équipe orange - constructeurs ayant appris des attaquants

Constructeurs qui **changent leur conception** et/ou **adaptent la mise en œuvre** de leurs solutions à partir de leurs **connaissances des techniques des attaquants**.

Équipe verte - constructeurs ayant appris des défenseurs

Constructeurs qui **intègrent les pratiques défensives** dans leurs produits.

Il est fréquent que d'autres équipes interviennent au niveau du pilotage des pentests :

Équipe blanche - observateurs

Équipe neutre jouant le rôle d'**arbitre** entre une équipe rouge de faux délinquants et une équipe bleue de véritables défenseurs.
L'équipe blanche établit des **règles d'engagement** (ROE) et des **mesures de performance** des tests de sécurité. Elle est également chargée de tirer les leçons apprises des tests dans une **évaluation post-engagement** qu'elle présente à la Direction.
En cas d'urgence, il arrive que l'équipe blanche participe aux activités de réponse aux incidents et attaques de robots.

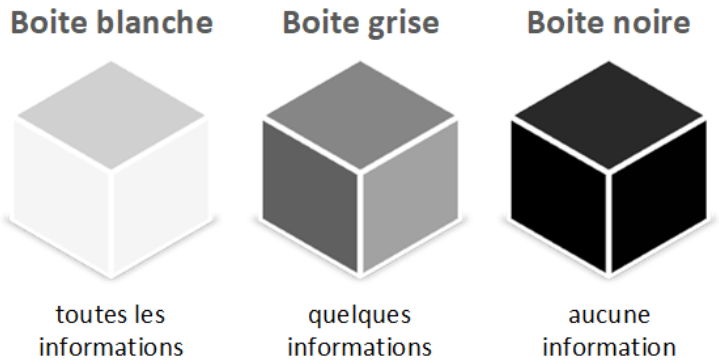
Équipe or - simulation de crise

Équipe chargée de la **simulation scénarisée d'une crise**, appelée également simulation de salle de guerre.

Équipe noire - opération secrète

Équipe généralement en charge de l'**évaluation de la sécurité physique** ; elle tire son nom d'opérations furtives durant lesquelles les attaquants s'habillaient tout en noir.
Les principales caractéristiques d'une opération noire sont qu'elle est **secrète** et **non imputable à priori à l'organisation** qui en est la **mandataire** (opération sous fausse bannière).

Pour terminer, il est nécessaire de définir le niveau d'informations dont disposerons les pentesters :



WhiteBox - boite blanche

Le testeur est en possession de nombreuses informations (schémas d'architecture, comptes utilisateur ou administrateur, code source applicatif, etc.) et n'a donc qu'à **rechercher les failles** et trouver le moyen de les exploiter. Généralement, étant à l'intérieur du réseau, avec toutes les ressources utiles, il aura beaucoup de facilité à trouver comment attaquer le système.

BlackBox - boite noire

Le testeur se met dans la peau d'un attaquant potentiel et **ne possède aucune information**.
Bien que sa capacité soit considérablement limitée par l'absence d'information d'identification, il dispose d'un certain nombre d'attaques et de tactiques lui permettant d'obtenir un accès ou des informations sensibles, par exemple :

- la découverte de contenu ;
- l'utilisation d'information par défaut ;
- l'énumération de noms utilisateur à partir d'un dictionnaire ;
- etc.

GreyBox - boite grise

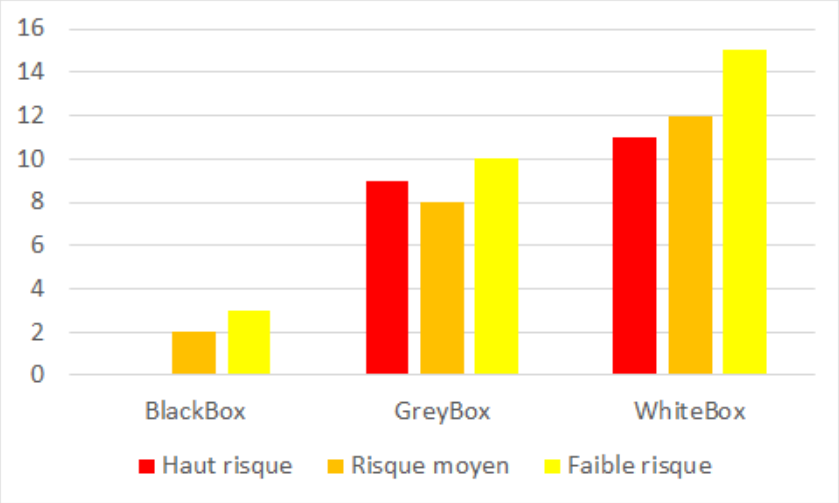
Méthode intermédiaire entre la WhiteBox et la BlackBox, souvent considérée comme une technique optimale qui permet de tester différents types d'attaques, tant internes qu'externes.
En général, le testeur dispose d'un couple identifiant / mot de passe uniquement, qui lui permet de passer l'étape d'authentification. Il évalue ensuite le niveau de sécurité du SI vis-à-vis d'un **utilisateur normal**. Il vérifiera notamment s'il lui est possible d'**acquérir des privilèges** au fur et à mesure de l'attaque.

Le budget d'une opération de pentest peut rapidement grimper, en fonction de l'organisation choisie :

BlackBox	GreyBox	WhiteBox
€ Tests les **moins chers**	€€ Coût **moyen**	€€€ Tests les **plus coûteux**
+ Découverte d'un **nombre très faible** de vulnérabilités	++ Découverte d'un **nombre substantiel** de vulnérabilités	+++ Découverte du **plus grand nombre** de vulnérabilités

La rentabilité des tests en WhiteBox est en réalité pondérée par le **temps élevé** nécessaire pour identifier des vulnérabilités - temps qu'il faut bien sûr rémunérer !

Pour illustrer cette réalité, prenons le nombre moyen de vulnérabilités découvertes par chaque type de pentest :



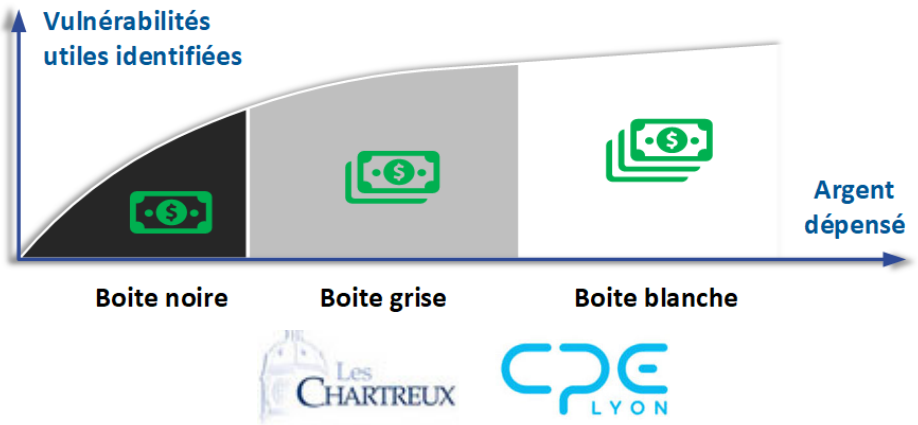
… nombre de vulnérabilités que l'on peut pondérer selon le niveau de risque :

	Pondération	BlackBox	GreyBox	WhiteBox
Haut risque	x 1	0	9	11
Risque moyen	x 0,5	1	4	6
Faible risque	x 0,25	0,75	2,5	3,75
Total		**1,75**	**15,5**	**20,75**

Ce qui nous permet d'établir le coût réel d'une vulnérabilité pour chaque type de pentest :

	BlackBox	GreyBox	WhiteBox
Coût moyen de l'opération de pentest	4 000 €	12 000 €	30 000 €
Nombre de vulnérabilités découvertes	1,75	15,5	20,75
Coût d'une vulnérabilité	**2 285 €**	**774 €**	**1 446 €**

Ainsi, le nombre de vulnérabilités découvertes augmente de manière significative au-delà de la BlackBox, mais la rentabilité des tests diminue sensiblement en WhiteBox :



Audit de code

L'audit de code est une pratique **longue** et **très fastidieuse** ayant pour objectif de vérifier la **sécurité du code d'une application** selon deux aspects :

- **l'aspect technique** : il s'agit de valider le respect des bonnes pratiques de développement associées à la production de code en général, et plus spécifiquement aux langages employés, avec une attention toute particulière portée aux éléments de sécurité intégrés à l'application ;
- **l'aspect fonctionnel** : il s'agit de valider la bonne implémentation des fonctionnalités et le respect des bonnes pratiques de conception associées, indépendamment des technologies utilisées.

En raison de la **complexité de l'exercice**, l'audit de code ne permet généralement pas de dresser une liste exhaustive des vulnérabilités du code.

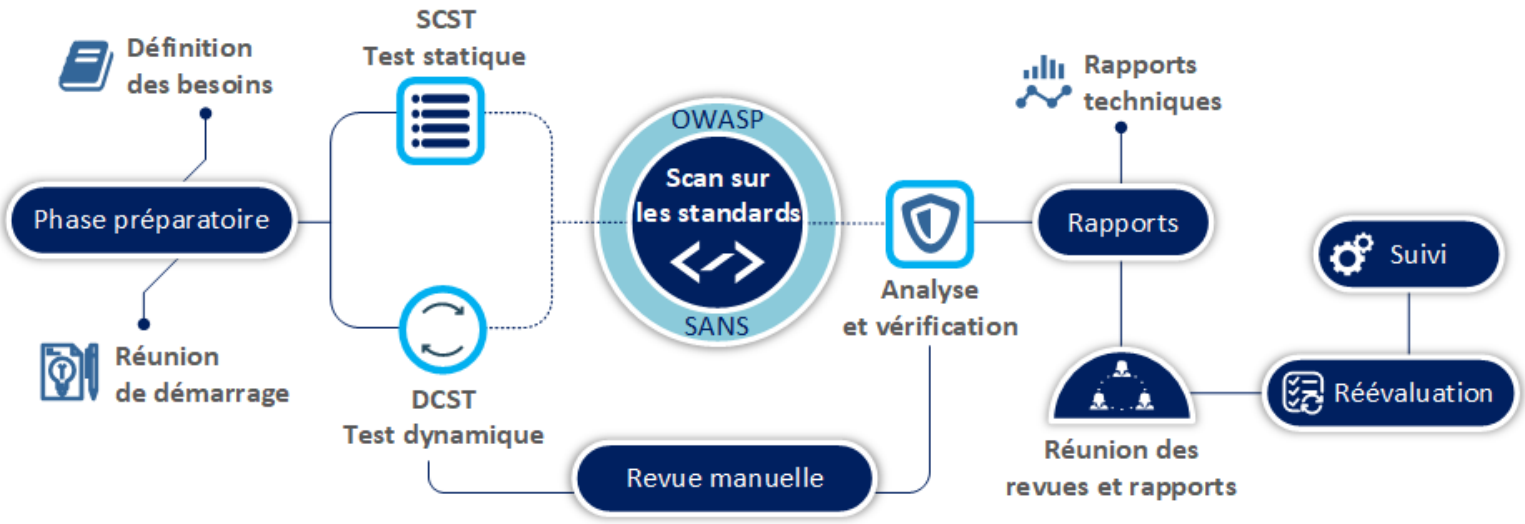
L'on effectue généralement ce genre d'audit :

- en cas de **manque de maitrise** d'un code, qui n'est compréhensible que par ceux qui l'ont développé ;
- lorsque l'on a des **doutes sur l'état de santé** de l'application, sa performance et/ou sa perméabilité ;
- afin de **mesurer le coût de maintenance** de l'application, directement dépendant de la quantité de lignes de code ;
- pour **anticiper d'éventuels problèmes** d'évolution applicative, notamment lorsque le code n'est pas bien structuré et documenté.
- dans le cas d'un applicatif **partiellement ou totalement externalisé**, que l'on ne gère pas ou pour lequel il est difficile d'obtenir des informations au quotidien.

Il existe deux grandes méthodologies d'audit de code source :

- **l'audit de code source statique (SCST)**,
  pour lequel l'auditeur se charge de **lire** et **comprendre chaque ligne de code**, afin d'y déceler des défauts de programmation ;
- **l'audit de code source dynamique (DCST)**,
  pour lequel l'auditeur utilise un outil d'analyse **automatique** qui trouve des failles de sécurités, tel que les **RATS**.
  Note : bien que la valeur de ce type d'outils soit certaine, ils ne font en réalité que le gros du travail et il est probable qu'ils passent à côté de problèmes qu'un œil humain verrait sans effort. Dans tous les cas, une **analyse statique complémentaire** est nécessaire pour **éliminer les faux positifs.**

Un audit de code source s'effectue généralement en suivant ce processus-type :



***OWASP** : Open Web Application Security Project*
***SANS** : administration, mise en réseau et sécurité du système*

Fuzzing

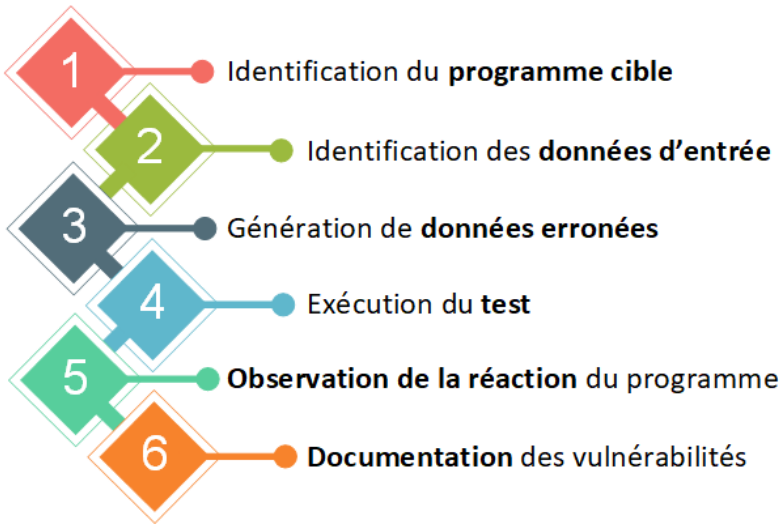
Dans le cadre d'un pentest BlackBox, le code n'est certes pas disponible, mais il existe une alternative à l'analyse de code, nommée **Fuzzing** ou **Fuzz Testing**. C'est une **méthode automatisée** permettant de mettre à jour les **vulnérabilités d'un logiciel** par l'analyse de son comportement face à l'injection en entrée de données plus ou moins aléatoires, avec des valeurs limites.
Aussi, le Fuzz Testing permet d'évaluer si les réactions attendues face aux différentes variantes de saisie existent bien dans le programme : les **saisies erronées** ou **défectueuses** doivent être compensées par des routines de traitement d'erreurs, sans quoi le programme plante généralement.

Contrairement à l'audit de code qui est une analyse structurelle, le Fuzzing est une **analyse comportementale** d'une application.

Il existe trois variantes de Fuzzing :

Fuzzing d'application	Fuzzing de protocoles	Fuzzing de format de fichiers
Test des **fonctions**, des **boutons** et des **champs de saisie** de programmes graphiques Test des **options de programmes** en lignes de commande ➢ Sollicitation des fonctions de manière ciblée, beaucoup **plus fréquemment** ou **rapidement** que d'ordinaire ➢ Injection de **contenus trop volumineux** dans les champs de saisie	Test d'un programme qui **interprète un protocole**, tel qu'HTTP ➢ Envoi de **contenus mal formatés** *Il est particulièrement important que les données erronées **ne soient pas interprétées comme des commandes** exécutées sur un serveur*	Test d'un programme qui **interprète un format de fichier** ➢ Demande de traitement d'un fichier au **format erroné** (éventuellement d'une fonction avancée comme la compression d'un fichier vidéo) *On testera en priorité des **formats normalisés**, tels que .jpg ou .pdf, car ce sont ceux utilisés dans les échanges entre applications.*

Un Fuzz Testing s'effectue généralement en suivant ce processus-type :



Conclusion

La **demande de services** de tests de pénétration et d'évaluation de la sécurité **augmente** d'année en année dans le monde, poussée par des préoccupations de Gouvernance, de Risque et de Conformité (**GRC**), ainsi qu'une **prise de conscience** quant aux enjeux liés à la sécurité et la confidentialité des données clients.
La plupart des responsables, tous secteurs confondus, doivent s'attendre à devoir effectuer des **évaluations régulières** pour valider la sécurité de leur organisation et de leurs produits.

Une circonstance malheureuse de deux décennies de développement et de professionnalisation du **pentesting** est que le terme couvre désormais une **large gamme** d'offres de sécurité, d'attributs de risque, de services différenciés, etc., avec des **termes inter mélangés**, des méthodologies particulières ou des test hybrides.
Déchiffrer le jargon et **identifier le bon prestataire** sont souvent décourageants car une erreur n'a pas qu'un impact financier, elle peut également mettre fin à une carrière si elle s'avère trop sérieuse.

Toutes les méthodologies d'audit s'appuient sur la représentation d'une menace sous l'angle d'un vecteur d'attaque ou d'une exploitation. Un différenciateur-clé réside dans leur stratégie : rechercher la **présence de vulnérabilités**, ou **exploiter des failles** à travers une attaque maitrisée.
La première sera généralement mise en évidence dans la diversité d'**évaluations et** d'**audits,** tandis que la seconde sera définie par le métier du **test de pénétration** (ou hack éthique).

Activité

Il est attendu le rendu de cette activité sous la forme d'un dossier déposé sur e-campus.
Vous rédigerez une analyse détaillée et justifiée des audits à mener dans chacune des situations proposées.

A. L'entreprise A souhaite maintenir la confiance de ses clients et de ses partenaires commerciaux. Elle aimerait s'assurer de l'intégrité de ses données et de son capital informationnel. Elle voudrait donc évaluer régulièrement ses mesures de sécurité et de contrôle.

B. L'entreprise B a été victime d'un ransomware. Suite à l'intervention d'un prestataire spécialisé en SSI, elle aimerait savoir si elle est bien préparée à une nouvelle future attaque.

C. L'entreprise C a été rachetée par une entreprise américaine cotée en bourse. Selon la législation, elle doit se conformer à la normalisation SOX ayant un impact direct avec le SI.

D. L'entreprise D a vu son RSSI démissionner après que l'on ait découvert qu'il avait caché un risque de sécurité majeur à la Direction. Cette dernière souhaite comprendre cette situation et y remédier avant la prise de poste d'un nouveau RSSI.

E. L'entreprise E développe une application en mode SaaS. L'ensemble de l'architecture est hébergé chez un prestataire et seuls les codes basés sur les frameworks du prestataire sont accessibles aux développeurs. N'ayant pas la main sur l'architecture logicielle, l'entreprise souhaite évaluer la robustesse de son application pour identifier s'il faut faire évoluer son prestataire sur des risques de sécurité.

F. L'entreprise F souhaite mettre en œuvre un règlement intérieur garantissant le bon usage de son SI par ses salariés. Elle voudrait identifier les usages actuels, élaborer son règlement puis vérifier que les usages futurs sont conformes.

G. L'entreprise G souhaite challenger le grand public sur son SI dans un *Bug Bounty*[1]. Avant de lancer le projet, elle aimerait évidement faire intervenir des professionnels pour réduire au maximum la perméabilité de son SI.

H. L'entreprise H souhaite établir une base documentaire de tous ses processus SI afin de les maintenir à jour.

I. L'entreprise I veut mettre en œuvre un nouveau logiciel dans son SI. Au préalable, il est nécessaire de mesurer son impact éventuel sur la PSSI.

J. L'entreprise J a financé plusieurs prestataires afin d'établir une sécurité avancée de son SI. Faisant partie des OIV, elle ambitionne d'aller plus loin sur le sujet de la cybersécurité : pour éprouver son système, elle souhaite simuler une situation de crise en ayant une analyse précise de chacun des acteurs parties prenantes du processus. Son objectif est d'identifier le maximum de vulnérabilités, tant techniques que comportementales.

K. L'entreprise K développe une application basée sur une architecture N/tiers Web. Elle aimerait se conformer aux bonnes pratiques de développement afin d'avoir la meilleure perspective d'évolution et de flexibilité possible.

[1] *Bug Bounty* : prime aux bogues (aussi appelée chasse aux bogues ; en anglais, bug bounty), c'est un programme de récompenses proposé par de nombreux sites web et développeurs de logiciels, qui offre des récompenses aux personnes qui rapportent des bogues, surtout ceux associés à des vulnérabilités.