

Cybersécurité, logiciel de vol et résilience opérationnelle

Protéger les fonctions numériques d'une base isolée: architecture de confiance, authentification, mises à jour, logiciels critiques, journaux, segmentation et [récupération](#) après compromission.

Comprendre le problème avant les équations

Question de départ — Cybersécurité, logiciel de vol et résilience opérationnelle: comment « Comprendre avant de calculer » agit-il sur « cybersécurité », et quel repère issu de « intégrité » permet de vérifier la décision?

Intuition. Un réseau de bord reçoit une commande apparemment valide pendant une phase critique. Il faut distinguer panne, erreur logicielle, corruption de données et action malveillante.

1

Comprendre avant de calculer

Un réseau de bord reçoit une commande apparemment valide pendant une phase critique.

2

cybersécurité

Définition sans prérequis.

3

intégrité

Définition sans prérequis.

Repère concret. Sécuriser uniquement le réseau externe laisse des risques dans logiciels, mises à jour, maintenance et accès physique.

- Comprendre avant de calculer — Un réseau de bord reçoit une commande apparemment valide pendant une phase critique.
- cybersécurité — Définition sans prérequis.
- intégrité — Définition sans prérequis.

Pour aborder « Cybersécurité, logiciel de vol et résilience opérationnelle », les prérequis conseillés sont modules 00 à 34 recommandés selon le sujet. Pour « Cybersécurité, logiciel de vol et résilience opérationnelle », une notation technique n'est utilisée qu'après avoir été reliée à son sens physique ou opérationnel.

RAMPE ZÉRO PRÉREQUIS · MODULE 38

Comprendre avant de calculer

Un réseau de bord reçoit une commande apparemment valide pendant une [phase](#) critique. Il faut distinguer panne, erreur logicielle, corruption de données et action malveillante.

Progression du module — cybersécurité; intégrité; authentification; mode sûr.

cybersécurité

Définition sans prérequis. La [cybersécurité](#) vise à protéger confidentialité lorsque nécessaire, intégrité, [disponibilité](#) et authenticité des systèmes numériques face aux erreurs et attaques.

Scène concrète. Une commande de vanne doit provenir d'une source autorisée et ne pas avoir été modifiée en transit.

Contrôle mental. Testez la défense par scénario de menace: qui peut agir, avec quel accès, quelle trace reste et comment le système continue en sécurité si la confiance est perdue.

Piège à éviter. Sécuriser uniquement le réseau externe laisse des risques dans logiciels, mises à jour, maintenance et accès physique.

Pourquoi cela compte en mission. Dans ce module, une erreur sur « cybersécurité » peut transformer une erreur logicielle ou une compromission en perte de commande, de données ou de capacité de récupération.

Exercice guidé — cybersécurité

Situation à reconnaître. Une commande de vanne doit provenir d'une source autorisée et ne pas avoir été modifiée en transit.

Vérification demandée. Testez la défense par scénario de menace: qui peut agir, avec quel accès, quelle trace reste et comment le système continue en sécurité si la confiance est perdue.

Erreur à écarter. Sécuriser uniquement le réseau externe laisse des risques dans logiciels, mises à jour, maintenance et accès physique.

Corrigé raisonné

Sens précis

La cybersécurité vise à protéger confidentialité lorsque nécessaire, intégrité, disponibilité et authenticité des systèmes numériques face aux erreurs et attaques.

Test du cas

Testez la défense par scénario de menace: qui peut agir, avec quel accès, quelle trace reste et comment le système continue en sécurité si la confiance est perdue.

Piège exclu

Sécuriser uniquement le réseau externe laisse des risques dans logiciels, mises à jour, maintenance et accès physique.

Conséquence opérationnelle

Dans ce module, une erreur sur « cybersécurité » peut transformer une erreur logicielle ou une compromission en perte de commande, de données ou de capacité de récupération.

Termes du corrigé. [vérification](#) · [observation](#)

intégrité

Définition sans prérequis. L'intégrité signifie que données, commandes ou logiciels n'ont pas été altérés de façon non autorisée ou non détectée.

Scène concrète. Un checksum ou une signature peut aider à détecter certaines modifications d'un fichier de configuration.

Contrôle mental. Un contrôle d'intégrité doit détecter une modification non autorisée des données ou du logiciel; une donnée disponible mais altérée n'est pas une donnée fiable.

Piège à éviter. Une donnée intègre peut néanmoins être fausse si le capteur lui-même fournit une mesure erronée.

Pourquoi cela compte en mission. Dans ce module, une erreur sur « intégrité » peut transformer une erreur logicielle ou une compromission en perte de commande, de données ou de capacité de récupération.

Exercice guidé — intégrité

Situation à reconnaître. Un checksum ou une signature peut aider à détecter certaines modifications d'un fichier de configuration.

Vérification demandée. Un contrôle d'intégrité doit détecter une modification non autorisée des données ou du logiciel; une donnée disponible mais altérée n'est pas une donnée fiable.

Erreur à écarter. Une donnée intègre peut néanmoins être fausse si le capteur lui-même fournit une mesure erronée.

Corrigé raisonné

Sens précis

L'intégrité signifie que données, commandes ou logiciels n'ont pas été altérés de façon non autorisée ou non détectée.

Test du cas

Un contrôle d'intégrité doit détecter une modification non autorisée des données ou du logiciel; une donnée disponible mais altérée n'est pas une donnée fiable.

Piège exclu

Une donnée intègre peut néanmoins être fausse si le capteur lui-même fournit une mesure erronée.

Conséquence opérationnelle

Dans ce module, une erreur sur « intégrité » peut transformer une erreur logicielle ou une compromission en perte de commande, de données ou de capacité de récupération.

Quantification

intégrité: relation qualitative ici; aucune unité intrinsèque.

Vérification

intégrité: confronter conclusion, contrôle et piège de la carte.

authentification

Définition sans prérequis. L'**authentification** vérifie l'identité d'un utilisateur, d'un équipement ou d'un service avant d'autoriser certaines actions.

Scène concrète. Un terminal de maintenance peut exiger une clé matérielle et un secret pour obtenir des privilèges élevés.

Contrôle mental. Vérifiez que l'identité est prouvée avant d'accorder l'autorisation: connaître un identifiant ne doit pas suffire à obtenir un privilège critique.

Piège à éviter. Authentifier quelqu'un ne signifie pas qu'il est autorisé à toutes les opérations: authentification et autorisation sont distinctes.

Pourquoi cela compte en mission. Dans ce module, une erreur sur « **authentification** » peut transformer une erreur logicielle ou une compromission en perte de commande, de données ou de capacité de récupération.

Exercice guidé — authentification

Situation à reconnaître. Un terminal de maintenance peut exiger une clé matérielle et un secret pour obtenir des privilèges élevés.

Vérification demandée. Vérifiez que l'identité est prouvée avant d'accorder l'autorisation: connaître un identifiant ne doit pas suffire à obtenir un privilège critique.

Erreur à écarter. Authentifier quelqu'un ne signifie pas qu'il est autorisé à toutes les opérations: authentification et autorisation sont distinctes.

Corrigé raisonné

Sens précis

L'authentification vérifie l'identité d'un utilisateur, d'un équipement ou d'un service avant d'autoriser certaines actions.

Test du cas

Vérifiez que l'identité est prouvée avant d'accorder l'autorisation: connaître un identifiant ne doit pas suffire à obtenir un privilège critique.

Piège exclu

Authentifier quelqu'un ne signifie pas qu'il est autorisé à toutes les opérations: authentification et autorisation sont distinctes.

Conséquence opérationnelle

Dans ce module, une erreur sur « authentification » peut transformer une erreur logicielle ou une compromission en perte de commande, de données ou de capacité de récupération.

Quantification

authentification: relation qualitative ici; aucune unité intrinsèque.

Vérification

authentification: confronter conclusion, contrôle et piège de la carte.

mode sûr

Définition sans prérequis. Un **mode sûr** est un **état** conçu pour préserver les fonctions essentielles lorsque la situation ou la connaissance du système devient insuffisante.

Scène concrète. Après détection d'une incohérence logicielle, un véhicule peut couper des charges non essentielles et maintenir communication, thermique et attitude.

Contrôle mental. Déclenchez mentalement la perte d'un capteur ou d'un logiciel: le **mode sûr** doit réduire les risques, préserver énergie/thermique/communication essentiels et attendre une reprise contrôlée.

Piège à éviter. Un mode sûr mal conçu peut devenir un piège s'il consomme une ressource limitée ou s'il est impossible d'en sortir sans une fonction elle-même indisponible.

Pourquoi cela compte en mission. Dans ce module, une erreur sur « **mode sûr** » peut transformer une erreur logicielle ou une compromission en perte de commande, de données ou de capacité de récupération.

Exercice guidé — mode sûr

Situation à reconnaître. Après détection d'une incohérence logicielle, un véhicule peut couper des charges non essentielles et maintenir communication, thermique et attitude.

Vérification demandée. Déclenchez mentalement la perte d'un capteur ou d'un logiciel: le mode sûr doit réduire les risques, préserver énergie/thermique/communication essentiels et attendre une reprise contrôlée.

Erreur à écarter. Un mode sûr mal conçu peut devenir un piège s'il consomme une ressource limitée ou s'il est impossible d'en sortir sans une fonction elle-même indisponible.

Corrigé raisonné

Sens précis

Un mode sûr est un état conçu pour préserver les fonctions essentielles lorsque la situation ou la connaissance du système devient insuffisante.

Test du cas

Déclenchez mentalement la perte d'un capteur ou d'un logiciel: le mode sûr doit réduire les risques, préserver énergie/thermique/communication essentiels et attendre une reprise contrôlée.

Piège exclu

Un mode sûr mal conçu peut devenir un piège s'il consomme une ressource limitée ou s'il est impossible d'en sortir sans une fonction elle-même indisponible.

Conséquence opérationnelle

Dans ce module, une erreur sur « mode sûr » peut transformer une erreur logicielle ou une compromission en perte de commande, de données ou de capacité de récupération.

Étude intégrée — Cybersécurité, logiciel de vol et résilience opérationnelle

Situation à analyser. Un réseau de bord reçoit une commande apparemment valide pendant une phase critique. Il faut distinguer panne, erreur logicielle, corruption de données et action malveillante.

- **cybersécurité:** Une commande de vanne doit provenir d'une source autorisée et ne pas avoir été modifiée en transit. Contrôle attendu: Testez la défense par scénario de menace: qui peut agir, avec quel accès, quelle trace reste et comment le système continue en sécurité si la confiance est perdue.
- **intégrité:** Un checksum ou une signature peut aider à détecter certaines modifications d'un fichier de configuration. Contrôle attendu: Un contrôle d'intégrité doit détecter une modification non autorisée des données ou du logiciel; une donnée disponible mais altérée n'est pas une donnée fiable.
- **authentification:** Un terminal de maintenance peut exiger une clé matérielle et un secret pour obtenir des privilèges élevés. Contrôle attendu: Vérifiez que l'identité est prouvée avant d'accorder l'autorisation: connaître un identifiant ne doit pas suffire à obtenir un privilège critique.
- **mode sûr:** Après détection d'une incohérence logicielle, un véhicule peut couper des charges non essentielles et maintenir communication, thermique et attitude. Contrôle attendu: Déclenchez mentalement la perte d'un capteur ou d'un logiciel: le mode sûr doit réduire les risques, préserver énergie/thermique/communication essentiels et attendre une reprise contrôlée.

Le diagnostic ne doit pas confondre « cybersécurité » et « intégrité ». Le premier se reconnaît ici par le cas suivant: Une commande de vanne doit provenir d'une source autorisée et ne pas avoir été modifiée en transit. Le second doit être contrôlé autrement: Un contrôle d'intégrité doit détecter une modification non autorisée des données ou du logiciel; une donnée disponible mais altérée n'est pas une donnée fiable.

Une seconde vérification croise « authentification » et « mode sûr ». Gardez comme alerte « Authentifier quelqu'un ne signifie pas qu'il est autorisé à toutes les opérations: authentification et autorisation sont distinctes. »; pour « mode sûr », utilisez plutôt ce test: Déclenchez mentalement la perte d'un capteur ou d'un logiciel: le mode sûr doit réduire les risques, préserver énergie/thermique/communication essentiels et attendre une reprise contrôlée.

Exercice de synthèse

Dans la situation de « Cybersécurité, logiciel de vol et résilience opérationnelle », classez les indices selon « cybersécurité » (Testez la défense par scénario de menace: qui peut agir, avec quel accès, quelle trace reste et comment le système continue en sécurité si la confiance est perdue.), « intégrité » (Un contrôle d'intégrité doit détecter une modification non autorisée des données ou du logiciel; une donnée disponible mais altérée n'est pas une donnée fiable.), « authentification » (Vérifiez que l'identité est prouvée avant d'accorder l'autorisation: connaître un identifiant ne doit pas suffire à obtenir un privilège critique.), « mode sûr »

(Déclenchez mentalement la perte d'un capteur ou d'un logiciel: le mode sûr doit réduire les risques, préserver énergie/thermique/communication essentiels et attendre une reprise contrôlée.) Décision suspendue si un contrôle échoue.

Corrigé de synthèse

Pour cybersécurité, Dans ce module, une erreur sur « cybersécurité » peut transformer une erreur logicielle ou une compromission en perte de commande, de données ou de capacité de récupération. Le garde-fou principal est: Sécuriser uniquement le réseau externe laisse des risques dans logiciels, mises à jour, maintenance et accès physique. Pour intégrité, Dans ce module, une erreur sur « intégrité » peut transformer une erreur logicielle ou une compromission en perte de commande, de données ou de capacité de récupération. Le garde-fou principal est: Une donnée intègre peut néanmoins être fausse si le capteur lui-même fournit une mesure erronée. Pour authentification, Dans ce module, une erreur sur « authentification » peut transformer une erreur logicielle ou une compromission en perte de commande, de données ou de capacité de récupération. Le garde-fou principal est: Authentifier quelqu'un ne signifie pas qu'il est autorisé à toutes les opérations: authentification et autorisation sont distinctes. Pour mode sûr, Dans ce module, une erreur sur « mode sûr » peut transformer une erreur logicielle ou une compromission en perte de commande, de données ou de capacité de récupération. Le garde-fou principal est: Un mode sûr mal conçu peut devenir un piège s'il consomme une ressource limitée ou s'il est impossible d'en sortir sans une fonction elle-même indisponible.

Mini-leçons quantitatives canoniques

Ce bloc constitue la référence quantitative du module; les valeurs d'exercice sont pédagogiques et doivent être remplacées par les données de mission avant toute décision réelle.

Charge de déploiement des correctifs

$$H_{\text{patch}} = n_{\text{assets}} \times t_{\text{patch}}$$

1 — Question concrète

Que permet de calculer « $H_{\text{patch}} = n_{\text{assets}} \times t_{\text{patch}}$ » dans « Charge de déploiement des correctifs » ?

2 — Intuition sans symboles

Le coût opérationnel d'un correctif dépend du nombre d'actifs et du temps sûr nécessaire pour chacun.

3 — Grandeurs

H_{patch} : charge totale de patch [h]; n_{assets} : actifs à corriger [asset]; t_{patch} : temps moyen par actif [h/actif]

4 — Formule

$$H_{\text{patch}} = n_{\text{assets}} \times t_{\text{patch}}$$

5 — Lecture

Lire « $H_{\text{patch}} = n_{\text{assets}} \times t_{\text{patch}}$ » en nommant chaque opération et chaque indice.

6 — Symboles et sens

H_{patch} : charge totale de patch [h]; n_{assets} : actifs à corriger [asset]; t_{patch} : temps moyen par actif [h/actif]

7 — Prononciation

Dire explicitement les indices, puissances, parenthèses et lettres grecques afin de reconstruire la relation sans voir la page.

8 — Unités

H_{patch} [h]; n_{assets} [asset]; t_{patch} [h/actif]

9 — Convention

Conserver le même repère, la même base de temps, la même frontière de système et la même convention de signe pendant toute la substitution.

10 — Pourquoi cette opération

La multiplication transforme un taux, une fraction ou une grandeur spécifique en quantité totale.

11 — Hypothèses

Les facteurs doivent appartenir au même scénario et leurs unités doivent se combiner correctement.

12 — Contrôle indépendant

Diviser le résultat par un facteur non nul doit retrouver le produit des autres.

13 — Cas numérique

Avec $n_{\text{assets}} = 20$ asset, $t_{\text{patch}} = 0.5$ h/actif: $H_{\text{patch}} = 20 \times 0.5 = 10$ h.

14 — Pourquoi le calcul fonctionne

La multiplication transforme un taux, une fraction ou une grandeur spécifique en quantité totale.

15 — Vérification

Diviser le résultat par un facteur non nul doit retrouver le produit des autres.

16 — Estimation mentale

Estimer séparément chaque facteur donne rapidement l'ordre de grandeur du produit.

17 — Interprétation

Planifier suffisamment de temps pour validation et rollback plutôt que de traiter le correctif comme une simple copie de fichier.

18 — Ce que le résultat ne prouve pas

Le calcul ne prouve ni que les entrées sont exactes ni que le modèle reste valide hors du scénario annoncé; ces deux points exigent des preuves séparées.

19 — Sensibilité

Une variation relative d'un facteur se transmet directement au produit si les autres restent constants.

20 — Exercices guidé et autonome

Exercice guidé. Recalculez ce scénario: Avec $n_{\text{assets}} = 50$ asset, $t_{\text{patch}} = 0.2$ h/actif: $H_{\text{patch}} = 50 \times 0.2$?

Correction guidée détaillée — ouvrir après essai

Avec $n_{\text{assets}} = 50$ asset, $t_{\text{patch}} = 0.2$ h/actif: $H_{\text{patch}} = 50 \times 0.2 = 10$ h. La décision doit ensuite être confrontée aux marges et hypothèses du module.

Exercice autonome. Recalculez ce scénario: Avec $n_{\text{assets}} = 12$ asset, $t_{\text{patch}} = 1.5$ h/actif: $H_{\text{patch}} = 12 \times 1.5$?

Correction autonome — ouvrir après essai

Avec $n_{\text{assets}} = 12$ asset, $t_{\text{patch}} = 1.5$ h/actif: $H_{\text{patch}} = 12 \times 1.5 = 18$ h. La décision doit ensuite être confrontée aux marges et hypothèses du module.

21 — Décision mission

Planifier suffisamment de temps pour validation et rollback plutôt que de traiter le correctif comme une simple copie de fichier.

Fenêtre d'exposition

$t_{\text{exposure}} = t_{\text{detect}} + t_{\text{validate}} + t_{\text{deploy}}$

1 — Question concrète

Que permet de calculer « $t_{\text{exposure}} = t_{\text{detect}} + t_{\text{validate}} + t_{\text{deploy}}$ » dans « Fenêtre d'exposition » ?

2 — Intuition sans symboles

L'exposition persiste pendant la détection, la validation et le déploiement; ces délais s'additionnent.

3 — Grandeurs

t_{exposure} : durée d'exposition [h]; t_{detect} : temps de détection [h]; t_{validate} : temps de validation [h]; t_{deploy} : temps de déploiement [h]

4 — Formule

$t_{\text{exposure}} = t_{\text{detect}} + t_{\text{validate}} + t_{\text{deploy}}$

5 — Lecture

Lire « $t_{\text{exposure}} = t_{\text{detect}} + t_{\text{validate}} + t_{\text{deploy}}$ » en nommant chaque opération et chaque indice.

6 — Symboles et sens

t_{exposure} : durée d'exposition [h]; t_{detect} : temps de détection [h]; t_{validate} : temps de validation [h]; t_{deploy} : temps de déploiement [h]

7 — Prononciation

Dire explicitement les indices, puissances, parenthèses et lettres grecques afin de reconstruire la relation sans voir la page.

8 — Unités

t_{exposure} [h]; t_{detect} [h]; t_{validate} [h]; t_{deploy} [h]

9 — Convention

Conserver le même repère, la même base de temps, la même frontière de système et la même convention de signe pendant toute la substitution.

10 — Pourquoi cette opération

L'addition rassemble des délais ou contributions qui s'accumulent dans la même chaîne opérationnelle.

11 — Hypothèses

Les contributions doivent être exprimées dans une unité commune et sur la même frontière de système.

12 — Contrôle indépendant

Retirer un terme du total doit restituer la somme des autres.

13 — Cas numérique

Avec $t_{\text{detect}} = 2$ h, $t_{\text{validate}} = 4$ h, $t_{\text{deploy}} = 3$ h: $t_{\text{exposure}} = 2 + 4 + 3 = 9$ h.

14 — Pourquoi le calcul fonctionne

L'addition rassemble des délais ou contributions qui s'accumulent dans la même chaîne opérationnelle.

15 — Vérification

Retirer un terme du total doit restituer la somme des autres.

16 — Estimation mentale

Additionner d'abord les termes dominants donne un ordre de grandeur robuste.

17 — Interprétation

Réduire en priorité le terme qui domine la fenêtre sans supprimer les contrôles de sûreté nécessaires.

18 — Ce que le résultat ne prouve pas

Le calcul ne prouve ni que les entrées sont exactes ni que le modèle reste valide hors du scénario annoncé; ces deux points exigent des preuves séparées.

19 — Sensibilité

Le total varie linéairement avec chaque terme si les autres restent constants.

20 — Exercices guidé et autonome

Exercice guidé. Recalculez ce scénario: Avec $t_{\text{detect}} = 0.5$ h, $t_{\text{validate}} = 2$ h, $t_{\text{deploy}} = 1$ h: $t_{\text{exposure}} = 0.5 + 2 + 1$?

Correction guidée détaillée — ouvrir après essai

Avec $t_{\text{detect}} = 0.5$ h, $t_{\text{validate}} = 2$ h, $t_{\text{deploy}} = 1$ h: $t_{\text{exposure}} = 0.5 + 2 + 1 = 3.5$ h. La décision doit ensuite être confrontée aux marges et hypothèses du module.

Exercice autonome. Recalculez ce scénario: Avec $t_{\text{detect}} = 8$ h, $t_{\text{validate}} = 6$ h, $t_{\text{deploy}} = 4$ h: $t_{\text{exposure}} = 8 + 6 + 4$?

Correction autonome — ouvrir après essai

Avec $t_{\text{detect}} = 8$ h, $t_{\text{validate}} = 6$ h, $t_{\text{deploy}} = 4$ h: $t_{\text{exposure}} = 8 + 6 + 4 = 18$ h. La décision doit ensuite être confrontée aux marges et hypothèses du module.

21 — Décision mission

Réduire en priorité le terme qui domine la fenêtre sans supprimer les contrôles de sûreté nécessaires.

Temps de restauration

$t_{\text{restore}} = t_{\text{isolate}} + t_{\text{restore_data}} + t_{\text{verify}}$

1 — Question concrète

Que permet de calculer « $t_{\text{restore}} = t_{\text{isolate}} + t_{\text{restore_data}} + t_{\text{verify}}$ » dans « Temps de restauration » ?

2 — Intuition sans symboles

Revenir au service exige isoler le défaut, restaurer un état connu et vérifier qu'il est réellement sain.

3 — Grandeurs

t_{restore} : temps total de restauration [h]; t_{isolate} : temps d'isolement [h]; $t_{\text{restore_data}}$: temps de restauration données [h]; t_{verify} : temps de vérification [h]

4 — Formule

$t_{\text{restore}} = t_{\text{isolate}} + t_{\text{restore_data}} + t_{\text{verify}}$

5 — Lecture

Lire « $t_{\text{restore}} = t_{\text{isolate}} + t_{\text{restore_data}} + t_{\text{verify}}$ » en nommant chaque opération et chaque indice.

6 — Symboles et sens

t_{restore} : temps total de restauration [h]; t_{isolate} : temps d'isolement [h]; $t_{\text{restore_data}}$: temps de restauration données [h]; t_{verify} : temps de vérification [h]

7 — Prononciation

Dire explicitement les indices, puissances, parenthèses et lettres grecques afin de reconstruire la relation sans voir la page.

8 — Unités

t_{restore} [h]; t_{isolate} [h]; $t_{\text{restore_data}}$ [h]; t_{verify} [h]

9 — Convention

Conserver le même repère, la même base de temps, la même frontière de système et la même convention de signe pendant toute la substitution.

10 — Pourquoi cette opération

L'addition rassemble des délais ou contributions qui s'accumulent dans la même chaîne opérationnelle.

11 — Hypothèses

Les contributions doivent être exprimées dans une unité commune et sur la même frontière de système.

12 — Contrôle indépendant

Retirer un terme du total doit restituer la somme des autres.

13 — Cas numérique

Avec $t_{\text{isolate}} = 1 \text{ h}$, $t_{\text{restore_data}} = 3 \text{ h}$, $t_{\text{verify}} = 2 \text{ h}$: $t_{\text{restore}} = 1 + 3 + 2 = 6 \text{ h}$.

14 — Pourquoi le calcul fonctionne

L'addition rassemble des délais ou contributions qui s'accumulent dans la même chaîne opérationnelle.

15 — Vérification

Retirer un terme du total doit restituer la somme des autres.

16 — Estimation mentale

Additionner d'abord les termes dominants donne un ordre de grandeur robuste.

17 — Interprétation

Comparer ce temps à l'autonomie du mode sûr avant d'accepter l'architecture de récupération.

18 — Ce que le résultat ne prouve pas

Le calcul ne prouve ni que les entrées sont exactes ni que le modèle reste valide hors du scénario annoncé; ces deux points exigent des preuves séparées.

19 — Sensibilité

Le total varie linéairement avec chaque terme si les autres restent constants.

20 — Exercices guidé et autonome

Exercice guidé. Recalculez ce scénario: Avec $t_{\text{isolate}} = 0.5 \text{ h}$, $t_{\text{restore_data}} = 1 \text{ h}$, $t_{\text{verify}} = 1 \text{ h}$: $t_{\text{restore}} = 0.5 + 1 + 1$?

Correction guidée détaillée — ouvrir après essai

Avec $t_{\text{isolate}} = 0.5 \text{ h}$, $t_{\text{restore_data}} = 1 \text{ h}$, $t_{\text{verify}} = 1 \text{ h}$: $t_{\text{restore}} = 0.5 + 1 + 1 = 2.5 \text{ h}$. La décision doit ensuite être confrontée aux marges et hypothèses du module.

Exercice autonome. Recalculez ce scénario: Avec $t_{\text{isolate}} = 2 \text{ h}$, $t_{\text{restore_data}} = 6 \text{ h}$, $t_{\text{verify}} = 3 \text{ h}$: $t_{\text{restore}} = 2 + 6 + 3$?

Correction autonome — ouvrir après essai

Avec $t_{\text{isolate}} = 2 \text{ h}$, $t_{\text{restore_data}} = 6 \text{ h}$, $t_{\text{verify}} = 3 \text{ h}$: $t_{\text{restore}} = 2 + 6 + 3 = 11 \text{ h}$. La décision doit ensuite être confrontée aux marges et hypothèses du module.

21 — Décision mission

Comparer ce temps à l'autonomie du mode sûr avant d'accepter l'architecture de récupération.

Couverture de sauvegarde vérifiée

$C_{\text{backup}} = n_{\text{verified}} / n_{\text{critical}}$

1 — Question concrète

Que permet de calculer « $C_{\text{backup}} = n_{\text{verified}} / n_{\text{critical}}$ » dans « Couverture de sauvegarde vérifiée » ?

2 — Intuition sans symboles

Une sauvegarde n'est une capacité que si sa restauration a été vérifiée pour les actifs critiques.

3 — Grandeurs

C_{backup} : couverture vérifiée [sans dimension]; n_{verified} : actifs avec restauration vérifiée [asset]; n_{critical} : actifs critiques [asset]

4 — Formule

$C_{\text{backup}} = n_{\text{verified}} / n_{\text{critical}}$

5 — Lecture

Lire « $C_{\text{backup}} = n_{\text{verified}} / n_{\text{critical}}$ » en nommant chaque opération et chaque indice.

6 — Symboles et sens

C_backup: couverture vérifiée [sans dimension]; n_verified: actifs avec restauration vérifiée [asset]; n_critical: actifs critiques [asset]

7 — Prononciation

Dire explicitement les indices, puissances, parenthèses et lettres grecques afin de reconstruire la relation sans voir la page.

8 — Unités

C_backup [sans dimension]; n_verified [asset]; n_critical [asset]

9 — Convention

Conserver le même repère, la même base de temps, la même frontière de système et la même convention de signe pendant toute la substitution.

10 — Pourquoi cette opération

La division rapporte une quantité à une capacité, une durée ou une référence cohérente.

11 — Hypothèses

Numérateur et dénominateur doivent décrire le même scénario et le dénominateur doit rester positif.

12 — Contrôle indépendant

Multiplier le résultat par le dénominateur doit reconstruire le numérateur.

13 — Cas numérique

Avec n_verified = 18 asset, n_critical = 20 asset: C_backup = 18 / 20 = 0.9 .

14 — Pourquoi le calcul fonctionne

La division rapporte une quantité à une capacité, une durée ou une référence cohérente.

15 — Vérification

Multiplier le résultat par le dénominateur doit reconstruire le numérateur.

16 — Estimation mentale

Comparer les ordres de grandeur du numérateur et du dénominateur permet d'anticiper le résultat.

17 — Interprétation

Traiter toute couverture incomplète comme une dette de résilience et non comme une simple tâche informatique.

18 — Ce que le résultat ne prouve pas

Le calcul ne prouve ni que les entrées sont exactes ni que le modèle reste valide hors du scénario annoncé; ces deux points exigent des preuves séparées.

19 — Sensibilité

Le résultat augmente avec le numérateur et diminue quand le dénominateur augmente.

20 — Exercices guidé et autonome

Exercice guidé. Recalculez ce scénario: Avec n_verified = 30 asset, n_critical = 30 asset: C_backup = 30 / 30 ?

Correction guidée détaillée — ouvrir après essai

Avec n_verified = 30 asset, n_critical = 30 asset: C_backup = 30 / 30 = 1 . La décision doit ensuite être confrontée aux marges et hypothèses du module.

Exercice autonome. Recalculez ce scénario: Avec n_verified = 9 asset, n_critical = 12 asset: C_backup = 9 / 12 ?

Correction autonome — ouvrir après essai

Avec n_verified = 9 asset, n_critical = 12 asset: C_backup = 9 / 12 = 0.75 . La décision doit ensuite être confrontée aux marges et hypothèses du module.

21 — Décision mission

Traiter toute couverture incomplète comme une dette de résilience et non comme une simple tâche informatique.

Précision des alertes

$P_{\text{alert}} = n_{\text{true}} / n_{\text{alerts}}$

1 — Question concrète

Que permet de calculer « $P_{\text{alert}} = n_{\text{true}} / n_{\text{alerts}}$ » dans « Précision des alertes » ?

2 — Intuition sans symboles

La précision mesure la part des alertes qui méritent réellement l'attention de l'équipage.

3 — Grandeurs

P_alert: précision des alertes [sans dimension]; n_true: alertes réellement pertinentes [alert]; n_alerts: alertes totales [alert]

4 — Formule

$$P_{\text{alert}} = n_{\text{true}} / n_{\text{alerts}}$$

5 — Lecture

Lire « $P_{\text{alert}} = n_{\text{true}} / n_{\text{alerts}}$ » en nommant chaque opération et chaque indice.

6 — Symboles et sens

P_{alert} : précision des alertes [sans dimension]; n_{true} : alertes réellement pertinentes [alert]; n_{alerts} : alertes totales [alert]

7 — Prononciation

Dire explicitement les indices, puissances, parenthèses et lettres grecques afin de reconstruire la relation sans voir la page.

8 — Unités

P_{alert} [sans dimension]; n_{true} [alert]; n_{alerts} [alert]

9 — Convention

Conserver le même repère, la même base de temps, la même frontière de système et la même convention de signe pendant toute la substitution.

10 — Pourquoi cette opération

La division rapporte une quantité à une capacité, une durée ou une référence cohérente.

11 — Hypothèses

Numérateur et dénominateur doivent décrire le même scénario et le dénominateur doit rester positif.

12 — Contrôle indépendant

Multiplier le résultat par le dénominateur doit reconstruire le numérateur.

13 — Cas numérique

Avec $n_{\text{true}} = 80$ alert, $n_{\text{alerts}} = 100$ alert: $P_{\text{alert}} = 80 / 100 = 0.8$.

14 — Pourquoi le calcul fonctionne

La division rapporte une quantité à une capacité, une durée ou une référence cohérente.

15 — Vérification

Multiplier le résultat par le dénominateur doit reconstruire le numérateur.

16 — Estimation mentale

Comparer les ordres de grandeur du numérateur et du dénominateur permet d'anticiper le résultat.

17 — Interprétation

Une précision trop faible crée de la fatigue et doit conduire à retuner la détection avant d'ajouter davantage d'alarmes.

18 — Ce que le résultat ne prouve pas

Le calcul ne prouve ni que les entrées sont exactes ni que le modèle reste valide hors du scénario annoncé; ces deux points exigent des preuves séparées.

19 — Sensibilité

Le résultat augmente avec le numérateur et diminue quand le dénominateur augmente.

20 — Exercices guidé et autonome

Exercice guidé. Recalculez ce scénario: Avec $n_{\text{true}} = 45$ alert, $n_{\text{alerts}} = 60$ alert: $P_{\text{alert}} = 45 / 60$?

Correction guidée détaillée — ouvrir après essai

Avec $n_{\text{true}} = 45$ alert, $n_{\text{alerts}} = 60$ alert: $P_{\text{alert}} = 45 / 60 = 0.75$. La décision doit ensuite être confrontée aux marges et hypothèses du module.

Exercice autonome. Recalculez ce scénario: Avec $n_{\text{true}} = 10$ alert, $n_{\text{alerts}} = 50$ alert: $P_{\text{alert}} = 10 / 50$?

Correction autonome — ouvrir après essai

Avec $n_{\text{true}} = 10$ alert, $n_{\text{alerts}} = 50$ alert: $P_{\text{alert}} = 10 / 50 = 0.2$. La décision doit ensuite être confrontée aux marges et hypothèses du module.

21 — Décision mission

Une précision trop faible crée de la fatigue et doit conduire à retuner la détection avant d'ajouter davantage d'alarmes.

Indice de risque opérationnel

$$R_{\text{oper}} = p_{\text{compromise}} \times I_{\text{impact}}$$

- 1 — Question concrète
Que permet de calculer « $R_{oper} = p_{compromise} \times I_{impact}$ » dans « Indice de risque opérationnel » ?
- 2 — Intuition sans symboles
Un indice simple combine probabilité et impact pour hiérarchiser des scénarios sans prétendre prédire exactement leur avenir.
- 3 — Grandeurs
 R_{oper} : indice de risque [point]; $p_{compromise}$: probabilité du scénario [sans dimension]; I_{impact} : impact normalisé [point]
- 4 — Formule
 $R_{oper} = p_{compromise} \times I_{impact}$
- 5 — Lecture
Lire « $R_{oper} = p_{compromise} \times I_{impact}$ » en nommant chaque opération et chaque indice.
- 6 — Symboles et sens
 R_{oper} : indice de risque [point]; $p_{compromise}$: probabilité du scénario [sans dimension]; I_{impact} : impact normalisé [point]
- 7 — Prononciation
Dire explicitement les indices, puissances, parenthèses et lettres grecques afin de reconstruire la relation sans voir la page.
- 8 — Unités
 R_{oper} [point]; $p_{compromise}$ [sans dimension]; I_{impact} [point]
- 9 — Convention
Conserver le même repère, la même base de temps, la même frontière de système et la même convention de signe pendant toute la substitution.
- 10 — Pourquoi cette opération
La multiplication transforme un taux, une fraction ou une grandeur spécifique en quantité totale.
- 11 — Hypothèses
Les facteurs doivent appartenir au même scénario et leurs unités doivent se combiner correctement.
- 12 — Contrôle indépendant
Diviser le résultat par un facteur non nul doit retrouver le produit des autres.
- 13 — Cas numérique
Avec $p_{compromise} = 0.1$ sans dimension, $I_{impact} = 8$ point: $R_{oper} = 0.1 \times 8 = 0.8$ point.
- 14 — Pourquoi le calcul fonctionne
La multiplication transforme un taux, une fraction ou une grandeur spécifique en quantité totale.
- 15 — Vérification
Diviser le résultat par un facteur non nul doit retrouver le produit des autres.
- 16 — Estimation mentale
Estimer séparément chaque facteur donne rapidement l'ordre de grandeur du produit.
- 17 — Interprétation
Utiliser l'indice pour prioriser les mitigations, puis réexaminer séparément les hypothèses de probabilité et d'impact.
- 18 — Ce que le résultat ne prouve pas
Le calcul ne prouve ni que les entrées sont exactes ni que le modèle reste valide hors du scénario annoncé; ces deux points exigent des preuves séparées.
- 19 — Sensibilité
Une variation relative d'un facteur se transmet directement au produit si les autres restent constants.
- 20 — Exercices guidé et autonome
Exercice guidé. Recalculez ce scénario: Avec $p_{compromise} = 0.02$ sans dimension, $I_{impact} = 10$ point: $R_{oper} = 0.02 \times 10$?

Correction guidée détaillée — ouvrir après essai

Avec $p_{compromise} = 0.02$ sans dimension, $I_{impact} = 10$ point: $R_{oper} = 0.02 \times 10 = 0.2$ point. La décision doit ensuite être confrontée aux marges et hypothèses du module.

Exercice autonome. Recalculez ce scénario: Avec $p_{compromise} = 0.3$ sans dimension, $I_{impact} = 4$ point: $R_{oper} = 0.3 \times 4$?

Correction autonome — ouvrir après essai

Avec $p_{compromise} = 0.3$ sans dimension, $I_{impact} = 4$ point: $R_{oper} = 0.3 \times 4 = 1.2$ point. La décision doit ensuite être confrontée aux marges et hypothèses du module.

- 21 — Décision mission

Progression de maîtrise

1. cybersécurité: appliquez le contrôle concret de sa fiche.
2. intégrité: appliquez le contrôle concret de sa fiche.
3. authentification: appliquez le contrôle concret de sa fiche.
4. mode sûr: appliquez le contrôle concret de sa fiche.
5. Synthèse de « Cybersécurité, logiciel de vol et résilience opérationnelle »: confrontez « cybersécurité » à « intégrité »; choisissez le contrôle prioritaire.

Vérification de maîtrise

Validation 38 — cybersécurité, intégrité, authentification, mode sûr. Pour chaque notion: un indice et un contrôle.

Objectifs de maîtrise

- Reconstruire avec ses propres mots le mécanisme étudié dans « Comprendre avant de calculer ».
- Vérifier par un **ordre de grandeur** le résultat utilisé dans « cybersécurité ».
- Proposer un contrôle indépendant avant de conclure sur « intégrité ».
- Justifier une action de mission à partir des limites observées dans « authentification ».

1. Sur Mars, une cyberattaque peut devenir une panne physique

Le logiciel commande énergie, air, eau, communications, robots et inventaires. Une action non autorisée ou une erreur logicielle peut donc avoir des effets matériels. La cybersécurité doit être intégrée à la sûreté de fonctionnement: protéger les données ne suffit pas si une commande falsifiée peut arrêter une pompe.

2. Réduire la surface d'attaque par l'architecture

Les systèmes critiques ne doivent pas tous partager le même réseau plat. Segmenter habitat, atelier, science, visiteurs, robots et administration limite la propagation d'une compromission. Les passerelles doivent autoriser uniquement les **flux** nécessaires et enregistrer les échanges sensibles.

3. Identité, authentification et moindre privilège

Un compte utilisateur ne doit pas posséder plus de droits que sa mission l'exige. Les opérations critiques peuvent demander authentification renforcée, double validation ou séparation des rôles. Le système doit aussi continuer à fonctionner lorsque la connexion avec la Terre est indisponible.

4. Mises à jour et chaîne de confiance

Un correctif logiciel utile peut devenir un risque s'il est corrompu ou incompatible. Les paquets doivent être signés, vérifiés, testés sur un environnement de référence puis déployés progressivement. Une image système connue et des versions antérieures doivent permettre le retour arrière.

5. Journalisation et détection sans noyer l'équipage

Les journaux doivent enregistrer commandes critiques, changements de configuration, échecs d'authentification et anomalies de communication. Trop d'alertes rend cependant l'outil inutile. La détection doit donc hiérarchiser les événements et conserver des données suffisantes pour reconstruire un incident.

6. Sauvegarde, mode sûr et restauration

Une base doit pouvoir repartir depuis des configurations propres hors ligne. Les sauvegardes doivent être séparées du système qu'elles protègent, testées et documentées. Certains automatismes critiques peuvent posséder un mode local minimal qui maintient pression, température ou circulation même si le réseau supérieur est indisponible.

7. Construire une frontière de confiance entre fonctions vitales et fonctions de confort

Tous les réseaux d'une base n'ont pas la même criticité. Un poste de divertissement, un serveur scientifique et le contrôleur d'une boucle d'oxygène ne devraient pas disposer des mêmes chemins ni des mêmes privilèges. La segmentation réduit le nombre de systèmes qu'un défaut logiciel ou une compromission peut atteindre. Mais elle doit rester compatible avec les opérations de secours: isoler un réseau ne doit pas empêcher un équipage autorisé d'accéder localement aux fonctions vitales. L'architecture doit donc définir zones de confiance, passerelles contrôlées, services indispensables, accès de maintenance et modes hors réseau. La cybersécurité devient une propriété de l'architecture fonctionnelle, pas une couche ajoutée après la mise en service.

8. Mettre à jour sans transformer un correctif en nouvelle panne

Une mise à jour logicielle peut corriger une vulnérabilité et introduire simultanément un défaut de compatibilité. Sur Mars, l'équipe ne peut pas compter sur un remplacement rapide du matériel ni sur l'intervention immédiate du fournisseur. Les correctifs doivent donc être authentifiés, testés sur une configuration représentative, installés par étapes et accompagnés d'un mécanisme de retour arrière. Les systèmes critiques peuvent maintenir une image logicielle connue comme sûre, séparée de la version courante. Les journaux doivent permettre de prouver quelle version tournait au moment d'un incident. Le patch management devient ainsi un processus de configuration et de sûreté autant qu'une activité de sécurité informatique.

9. Réponse à incident: préserver la mission avant de chercher le coupable

Lorsqu'un comportement suspect apparaît, la priorité opérationnelle est de maintenir les fonctions vitales et empêcher la propagation. L'équipe doit pouvoir isoler une machine, basculer vers un [mode dégradé](#), préserver les journaux, vérifier l'intégrité des commandes puis reconstruire le service depuis une base de confiance. Une analyse forensique détaillée peut venir ensuite. Dans un système distant, la réponse à incident doit être répétée en simulation: qui décide l'isolement, quelles fonctions peuvent être coupées, quels éléments doivent rester disponibles et comment confirmer que la restauration est saine? La procédure doit fonctionner même avec une liaison Terre indisponible.

10. Exemple calculé: budget de correctifs

Supposons 120 équipements logiciels. Une mise à jour demande 25 min de préparation et vérification par équipement. En séquence, cela représente 3 000 min = 50 h de travail. Si l'on automatise 80 % des vérifications et que l'opérateur ne consacre plus que 5 min par équipement, la charge tombe à 10 h. L'automatisation doit toutefois produire des preuves et des journaux, pas simplement masquer le travail.

Approfondissement: patcher un système vital sans fabriquer une panne commune

Un correctif logiciel peut réduire une vulnérabilité et créer simultanément un risque opérationnel. Sur Mars, la stratégie de mise à jour doit donc séparer les fonctions vitales, contrôler l'origine des paquets et prévoir un retour vers la dernière configuration connue comme stable. Les référentiels [NASA](#) de logiciel imposent des pratiques de planification, de sécurité et de contrôle de configuration précisément parce qu'un changement de code est aussi un changement du système physique qu'il commande. [NASA NPR 7150.2D](#)

Exemple calculé. Un habitat comporte 40 contrôleurs identiques. Mettre à jour les 40 simultanément expose potentiellement 100 % de cette famille à une régression commune. Une stratégie en quatre vagues de 10 contrôleurs ne change pas la probabilité intrinsèque qu'un paquet contienne un défaut, mais elle limite le rayon d'exposition initial à $10 \div 40 = 25$ %. Si la première vague fonctionne pendant une [période](#) d'observation définie, la suivante peut commencer. Pour une fonction réellement vitale, on ferait

encore mieux: versions redondantes indépendantes, banc de test représentatif et capacité de rollback vérifiée avant le déploiement.

Le chiffre de 25 % n'est pas une recommandation universelle. Il illustre la différence entre risque individuel et risque de [cause commune](#). Une mise à jour parfaitement authentifiée peut être dangereuse si elle installe le même défaut partout. La résilience cyber d'une colonie dépend donc autant de la segmentation, de la diversité et de la restauration que du chiffrement ou des mots de passe. [NASA — Spacecraft Software Engineering](#)

11. Exercice progressif

Dessinez un réseau comportant [ECLSS](#), production électrique, laboratoire, robots et réseau personnel. Indiquez quelles communications sont nécessaires et lesquelles doivent être bloquées par défaut.

Correction raisonnée

Une architecture acceptable sépare au minimum le réseau opérationnel vital du réseau scientifique et du réseau personnel. ECLSS et distribution électrique échangent les quelques états nécessaires à la gestion de puissance, mais n'acceptent pas de connexion entrante depuis les appareils personnels. Le laboratoire reçoit l'heure, la [télémétrie](#) autorisée et peut déposer ses résultats dans une zone intermédiaire; il ne commande pas directement les automates de vie. Les robots utilisent une passerelle authentifiée avec liste de commandes minimale. Le réseau personnel accède à des services publiés, jamais aux bus de contrôle. Toute exception est explicitement documentée, journalisée et limitée en protocole, direction et identité: « bloqué par défaut, ouvert par besoin démontré » devient la règle de conception.

Mini-projet

Écrire le plan de réponse à compromission d'un serveur de maintenance: détection, isolement, continuité des fonctions vitales, collecte de preuves, restauration, validation et retour en service.

Sources primaires et passerelles

- [NASA Software Engineering Handbook — Cybersecurity](#)
- [NASA NPR 7150.2D §3.11 — SWE-154, SWE-156, SWE-157, SWE-159 \(cybersécurité logicielle\)](#)
- [NASA JSC — Spacecraft Software Engineering](#)

Matrice de parité FR ↔ EN

Charge de déploiement des correctifs	Patch deployment workload
Fenêtre d'exposition	Exposure window
Temps de restauration	Restoration time
Couverture de sauvegarde vérifiée	Verified backup coverage
Précision des alertes	Alert precision
Indice de risque opérationnel	Operational risk index

Termes de la progression. [logiciel de vol](#) · [résilience](#)