Projet de programmation sécurisée

Objectifs du projet

Dans ce projet, vous travaillerez en équipe (au plus 3 étudiants) ou individuellement pour concevoir et mettre en œuvre un système logiciel qui possède une fonctionnalité de sécurité non triviale.

Éléments requis

Les éléments de sécurité suivants doivent être inclus dans chaque projet :

Autorisation – Le système doit appliquer une politique d'autorisation non triviale pour contrôler un sous-ensemble de son fonctionnement. Certains systèmes nécessiteront un DAC (Discretionary Access Control), d'autres un MAC (Mandatory Access Control). Mettez en œuvre un mécanisme de contrôle d'accès qui est approprié et naturel pour la fonctionnalité de votre système et l'échelle prévue.

Authentification – Le système doit authentifier ses clients ou permettre à ses clients d'authentifier le système. Les clients peuvent être des utilisateurs humains ou des programmes exécutés sur des machines. Vous devez mettre en œuvre des moyens d'authentification raisonnables, qui peuvent inclure des mots de passe, l'enregistrement des utilisateurs, la génération de secrets, la distribution de clés, etc.

Audit – Le système doit fournir une infrastructure pour l'audit ou d'autres moyens d'établir la responsabilité des actions.

Confidentialité et intégrité – Le système doit concerner des informations qui sont stockées à long terme ou qui sont transmises sur un réseau. La mission du système doit exiger que ces informations restent secrètes et soient protégées contre toute modification non autorisée.

En plus de ces éléments de sécurité, notez que tout le code du système doit être le vôtre, afin que vous puissiez évaluer ses propriétés de sécurité et donner une assurance à son comportement. Les exceptions suivantes à cette règle générale sont autorisées :

- Fonctionnalités fournies par la bibliothèque standard de votre langage (à l'exclusion des Framework de services Web) ;
- Bibliothèque de cryptographie ;
- Constructeur d'interface graphique ;

- Base de données ;
- Autres exceptions après discussion préalable avec Doudou-sensei.

Ces deux éléments ne sont pas nécessaires dans ce projet :

Disponibilité – De nombreuses techniques pour atteindre la disponibilité sont correctement étudiées dans le cadre d'autres cours, tels que les cours de réseaux, de bases de données, système d'exploitation, etc.
Interface utilisateur – Les interfaces utilisateurs fantaisistes (ou, en fait, toute interface utilisateur) ne sont pas requises pour ce projet. Si vous choisissez de développer une interface graphique, veillez à ce que son développement ne vous fasse pas perdre de temps sur les éléments de sécurité, qui sont la grande priorité de ce cours. La convivialité est plus importante que l'esthétique.

Sujets du projet

Les éléments de sécurité requis énumérés ci-dessus peuvent se manifester dans une variété de systèmes différents. Choisissez un système à mettre en œuvre dans la liste d'exemples ci-dessous, ou proposez votre propre système.

- **Programme de chat** – Permet aux utilisateurs d'envoyer des messages privés (texte, images/vidéos ?) avec d'autres utilisateurs. (Exemple : Signal)
- **Réseau social** – Permet aux utilisateurs de former des liens dans un réseau social et de partager des informations avec d'autres utilisateurs.
- **Stockage distribué** – Permet aux utilisateurs de partager et de synchroniser leurs fichiers sur plusieurs appareils informatiques. Il est de la plus haute importance de garantir l'intégrité et la confidentialité des données.
    - Au lieu de fichiers, des concepts similaires pourraient être mis en œuvre sous la forme d'un gestionnaire de mots de passe ou d'un porte-monnaie électronique pour des crypto-monnaies, synchronisant les données sur plusieurs appareils informatiques.
- **Vote électronique** – Permettre aux citoyens de voter, puis de recueillir et de comptabiliser leurs votes. Il est de la plus haute importance de garantir le secret du vote et de résister à la falsification.
- **Logiciel interne de banque** – Permettre aux clients d'exploiter un système bancaire à succursales multiples réparties dans une région géographique. Il est de la plus haute importance de veiller à ce que les comptes ne soient modifiés que par les parties autorisées.

- **Logiciel de dossiers médicaux** – Permettre aux clients (médecins) d'une organisation de soins de santé multisite de stocker et de mettre à jour les dossiers médicaux des patients.
- Ou autre système de votre propre conception...

Partie 1 – Charte

Votre charte de projet définit les membres de votre équipe (le cas échéant) et propose le système logiciel que vous avez l'intention de développer. Votre charte doit comprendre les informations suivantes :

- Un nom de travail pour votre système.
- Le langage d'implémentation que vous proposez d'utiliser, ainsi qu'une brève description de votre expérience antérieure dans ce langage.
- Un lien vers le dépôt de contrôle de version que vous utiliserez pour le développement du logiciel. (Bitbucket, GitHub, etc.…)
- Les membres de votre équipe et leurs coordonnées.
- Votre plan pour une réunion de groupe hebdomadaire. Il est fortement suggéré que vous vous rencontriez, en tant qu'équipe, au moins une fois par semaine en dehors des cours pour discuter de l'état d'avancement de votre projet.

Proposition :

- Fournissez un énoncé très court (1 paragraphe) de la vision ou de l'idée clé de votre système.
- Fournissez une courte liste à puces (pas plus de six éléments) des principales caractéristiques du système.
- Fournissez une liste à puces des éléments de sécurité essentiels et une description en une phrase de la manière dont votre système intégrera chaque élément.
- Fournissez une description narrative (3 à 4 paragraphes) du système que vous avez l'intention de concevoir.

Soumission : déposez un document PDF sur https://fad.esp.sn.

Évaluation – Le document de la Charte sera évalué en fonction des critères suivants :

- Le document doit être facile à comprendre, bien organisé et clairement écrit. Le style et l'usage corrects du français comptent ici, tout comme dans le monde réel.
- Les caractéristiques clés énumérées couvrent-elles l'ensemble des caractéristiques souhaitées (et non les détails) ?

- La description narrative est-elle suffisamment claire et complète pour permettre à un groupe différent de commencer à travailler sur votre projet et de réaliser votre vision ?

Partie 2 – Document sur les exigences

Votre document sur les exigences fournit une proposition complète pour le système que votre équipe va développer. Il doit contenir les cinq sections suivantes :

(1) **Objectif du système**. En 3 ou 4 paragraphes, expliquez l'objectif de votre système. La base de cette explication doit être le résumé d'une page du système que vous avez rédigé dans la partie 1. Mettez ce résumé à jour si nécessaire, en fonction des exigences fonctionnelles et des objectifs de sécurité que vous développez dans ce document.

(2) **Exigences fonctionnelles**. Établissez une liste d'exigences fonctionnelles pour le système que vous concevez. Pour les besoins de ce projet, une exigence fonctionnelle comprend les données suivantes :

a. Type d'utilisateur – Le type d'utilisateur concerné par cette exigence.
b. Actifs – Les informations critiques impliquées dans cette exigence.
c. Importance – L'importance de cette exigence pour l'objectif de votre système. Utilisez l'échelle de la méthode MoSCoW pour classer l'importance.
d. User story - Une brève description d'un seul type d'interaction qu'un utilisateur peut avoir avec votre système. Les user stories décrivent ce que votre système va faire, mais pas comment. Notez que les stories ne doivent pas porter sur les développeurs ou les attaquants, cela viendra plus tard.

Il peut être utile d'organiser vos exigences dans une feuille de calcul, afin de pouvoir les trier de différentes manières.

Les bonnes user stories répondent à la grille **INVEST** du développement agile de logiciels :

- Independant – Les stories ne devraient pas se chevaucher et, idéalement, vous devriez pouvoir les mettre en œuvre dans n'importe quel ordre, bien que certains ordres puissent avoir plus de sens que d'autres.
- Negociable – Les stories peuvent être amenées à changer au fur et à mesure de l'évolution de votre projet.
- Valuable – Les stories ajoutent de la valeur à votre système pour les utilisateurs.
- Estimable – L'importance des stories peut être estimée, donc classée. Et le temps de mise en œuvre des stories peut être estimé. Si vous ne pouvez pas estimer, vous ne comprenez

probablement pas l'histoire. Envisagez d'ajouter plus de détails ou de la diviser en histoires plus petites.

- Small – Les stories doivent être succinctes, probablement pas plus d'une à quatre phrases.
- Testable – Les stories représentent des objectifs testables. Si vous ne pouvez pas décider si vous avez atteint un objectif – c'est-à-dire si vous ne pouvez pas écrire un test qu'une personne extérieure qualifiée pourrait suivre pour déterminer si votre système satisfait une story – alors votre story n'est pas assez claire.

(3) **Analyse des menaces** – Identifiez les menaces qui pèsent sur votre système. Concentrez-vous principalement sur les menaces malveillantes et humaines. Contre quels types d'attaquants votre système se défendra-t-il ? Quelles sont leurs motivations, leurs ressources et leurs capacités ? Ne vous contentez pas d'énumérer des menaces vagues et génériques. Faites en sorte qu'elles soient spécifiques à votre système et à ses fonctionnalités. S'il existe des non-menaces, vous devez également les identifier. Par exemple, vous pouvez supposer que certains composants du système s'exécutent sur du matériel situé dans une salle des machines physiquement sécurisée, accessible uniquement par des opérateurs système dignes de confiance.

(4) **Objectifs de sécurité** – Identifiez les actifs et les parties prenantes impliqués dans votre système. Cette étape devrait être facile, car vous avez déjà identifié les actifs et les utilisateurs pour chaque exigence fonctionnelle. Pour chaque actif, identifiez sa valeur pour les parties prenantes. Effectuez une analyse des préjudices sur les biens. Utilisez le modèle "l'exécution d'une action sur/vers/avec un bien pourrait causer un préjudice". (Bien que vous soyez encouragé à réécrire les déclarations faites avec ce modèle dans un français plus naturel). Soyez aussi complet et créatif que possible ; c'est l'étape où vous risquez le plus de négliger un élément important et pertinent pour la sécurité. Transformez les préjudices que vous avez identifiés en objectifs de sécurité, en utilisant le modèle "le système doit empêcher toute action sur/vers/avec le bien". Attribuez à chaque objectif l'une des trois caractéristiques suivantes : confidentialité, intégrité ou disponibilité. Examinez la faisabilité de chaque objectif à la lumière de votre analyse des menaces. Si nécessaire, assouplissez les objectifs pour qu'il soit possible de les atteindre. Vous devez documenter toutes ces étapes, en indiquant vos actifs et vos parties prenantes, l'analyse des préjudices, l'analyse de faisabilité et les objectifs de sécurité qui en résultent.

Soumission : déposez un document PDF sur https://fad.esp.sn.

Évaluation : le document sur les exigences sera évalué en fonction des critères suivants :

- Votre document est-il facile à comprendre, bien organisé et clairement rédigé ? Le style et l'usage corrects du français comptent ici, tout comme dans le monde réel.
- L'objectif du système donne-t-il un résumé clair du système que vous développez ?
- Les exigences fonctionnelles semblent-elles judicieuses compte tenu de l'objectif du système ?
- Les exigences fonctionnelles sont-elles claires et semblent-elles complètes ?
- Les exigences fonctionnelles sont-elles conformes à la grille INVEST ?
- Les menaces et leurs caractéristiques sont-elles clairement décrites ?
- Est-ce que vous avez documenté toutes les étapes de l'élaboration de vos objectifs de sécurité, y compris vos analyses de préjudice et de faisabilité ?
- Est-ce que vous avez correctement identifié chaque objectif de sécurité comme étant lié à la confidentialité, l'intégrité ou la disponibilité ?
- Est-ce que les objectifs de sécurité que vous avez identifiés sont raisonnables par rapport à l'objectif du système, à la fonctionnalité prévue et aux menaces qui vous préoccupent ?
- Vous avez réussi à démontrer que la majorité des éléments de sécurité sont impliqués dans votre système.

Partie 3 et Partie 4 – Sprints !

Les parties 3 et 4 du projet impliquent le même travail essentiel. Vous vous engagerez dans un sprint, qui est une période fixe pendant laquelle vous développerez un incrément "livrable" de votre système. Vos sprints dureront trois semaines. À la fin de chaque sprint, vous expédierez le dernier incrément de votre système et en ferez la démonstration à votre classe. Vous disposez de deux cycles de sprint - nommés Alpha et Beta - pour mettre en œuvre votre projet.

Fonctionnalités – Vous êtes libre de décider des fonctionnalités que vous livrerez dans le cadre de chaque sprint. Mais le sprint Alpha doit démontrer les fonctionnalités liées à l'audit, tandis que le sprint Beta se concentre sur les fonctionnalités d'authentification et d'autorisation.

Contrôle de la source – Vous êtes tenu de conserver votre projet dans un système de contrôle de la source. Il existe de nombreuses possibilités gratuites (GitHub, Bitbucket, ...) qui fournissent un hébergement de code, un suivi des bogues et des wikis. En aucun cas vous ne devez avoir recours à l'envoi par e-mail de l'ensemble de votre code source entre les membres de l'équipe.

Assurance – Vous devez préparer un document qui explique comment, et ce que vous avez testé dans le code source soumis. Les points à discuter sont les suivants :

- Comment vous avez testé les méthodes, les classes et les unités individuelles ?

- Comment avez-vous testé l'intégration des unités, en particulier celles écrites par différents programmeurs ?
- Comment avez-vous testé les fonctionnalités que vous proposez ?
- Comment avez-vous testé pour vous assurer que les exigences de sécurité sont respectées ?
- Si vous avez fait de la programmation en binôme ou des revues de code.
- La couverture du code obtenue par votre suite de tests.
- La fréquence d'exécution des tests de régression.

Outil d'analyse statique – Vous êtes tenu d'utiliser un outil d'analyse statique pour évaluer votre code.

Démonstration – Pendant le cours, vous ferez une démonstration de votre programme le jour où vous devez le rendre. Apportez des ordinateurs portables sur lesquels votre système est installé et fonctionne. Doudou-sensei et les autres équipes de projet utiliseront votre système et feront un rapport sur la façon dont il semble fonctionner et atteindre ses objectifs de sécurité. Les démonstrations impliquent généralement une partie ou la totalité des événements suivants, bien qu'il y ait nécessairement une certaine flexibilité :

- Vous faites une démonstration de quelques user stories (exigences fonctionnelles) que vous avez marquées comme terminées. Tout bogue observé pendant cette démonstration serait particulièrement peu impressionnant. Doudou-sensei ou vos camarades de classe pourraient essayer des entrées bizarres, un comportement étrange de l'utilisateur, etc. pour tenter de trouver de tels bogues.
- Vous présentez le code source des fonctionnalités que vous livrez. La qualité du code est importante.
- Vous discutez de la conception et de la mise en œuvre de toute fonctionnalité de sécurité que vous livrez.
- Doudou-sensei et vos camarades de classe vous posent des questions. Les questions relatives à la sécurité de votre système peuvent être adressées à un membre spécifique du groupe. Tous les membres doivent être en mesure d'expliquer et de défendre les choix relatifs à la sécurité faits par votre groupe.

Soumission – Déposez les éléments suivants sur http://fad.esp.sn :

- Une version mise à jour de votre document sur les exigences, indiquant clairement les exigences qui ont été mises en œuvre et celles qui sont encore en attente ;
- Un document d'assurance décrivant votre approche et les tâches accomplies à ce jour ;

- Un fichier .zip contenant un exécutable compilé ;
- Un document PDF avec des instructions d'installation et d'exécution ;
- Lien vers votre dépôt de contrôle de version.

Partie 5 – Soumission finale

Soumettez une version mise à jour de votre document sur les exigences qui sert de rapport final. Dans ce document, ajoutez une discussion sur la façon dont votre conception répond à ces exigences :

- Autorisation : quels types de politiques d'autorisation sont utilisés (DAC vs. MAC) ? Comment sont-elles mises en œuvre ? Comment sont gérées les modifications des politiques ?
- Authentification : comment les utilisateurs et les programmes sont-ils authentifiés ?
- Audit : Comment la responsabilité des actions des utilisateurs est-elle assurée ?
- Confidentialité : comment la confidentialité des données est-elle assurée ?
- Intégrité : comment l'intégrité des données est-elle maintenue ?
- Comment les secrets (par exemple, les mots de passe) sont-ils choisis/générés, stockés, mis à jour et effacés ? Vous aurez probablement mis en œuvre une certaine utilisation de la cryptographie, il serait donc approprié de discuter de questions telles que la façon dont vous générez et effacez les clés, comment et où les clés sont stockées, les algorithmes cryptographiques utilisés, les tailles des clés et des nonces, et les hypothèses sur la fonctionnalité du système d'exploitation.
- Assurance : comment et qu'avez-vous testé dans le code source soumis ? Les questions à aborder sont les suivantes : comment avez-vous testé les méthodes, les classes et les unités individuelles ; comment avez-vous testé l'intégration des unités, en particulier celles écrites par des programmeurs différents ; comment avez-vous testé les fonctionnalités que vous livrez ; comment avez-vous testé pour vous assurer que les exigences de sécurité sont satisfaites ; si vous avez fait de la programmation en binôme ou des revues de code ; quelle couverture de code votre suite de tests atteint-elle ; et à quelle fréquence exécutez-vous les tests de régression.

Évaluation – La conception et la mise en œuvre de votre système, y compris votre code, seront évaluées en fonction de la façon dont elles illustrent les différents principes de sécurité, de la mesure dans laquelle votre système remplit son objectif, réalise ses objectifs de sécurité contre les menaces préoccupantes et implique les éléments de sécurité essentiels. La lisibilité, la simplicité, la documentation et les tests seront inclus dans cette évaluation.

Évaluation du projet

La note de votre projet sera calculée comme suit :

- Partie 1 – Document de la charte : 5%
- Partie 2 – Document sur les exigences : 20%.
- Partie 3 – Sprint Alpha (code et démo) : 10%
- Partie 4 – Sprint Beta (code et démo) : 20%
- Soumission finale : 45%

Soumission

Soumettez tous les fichiers sur le site https://fad.esp.sn. Une seule soumission est nécessaire pour un groupe donné.

Il y aura des soumissions séparées pour :

- Partie 1 – Charte du projet (soumettre en PDF) [05 mars 2024, 23h59 (UTC)]
- Partie 2 – Document sur les exigences du projet (soumettre en PDF) [26 mars 2024, 23h59 (UTC)]
- Partie 3 – Sprint Alpha [09 avril 2024, 23h59 (UTC)]
- Partie 4 – Sprint Beta [23 avril 2024, 23h59 (UTC)]
- Code final du projet (soumettre le lien vers le dépôt de contrôle de version, le rapport final et déposer le binaire compilé). [07 mai 2024, 23h59 (UTC)]