Відкритий проект
захисту веб-додатків

Стандарт оцінювання відповідності безпеки додатків 3.0

Жовтень 2015 р.

Зміст

Подяка

Версія 3.0, 2015 р.

Керівники проекту	Провідні автори	Співавтори та рецензенти
Ендрю ван дер Сток (*Andrew van der Stock*) Даніель Кусберт (*Daniel Cuthbert*)	Джим Маніко (*Jim Manico*)	Бой Баукема (*Boy Baukema*) Арі Кезеніемі (*Ari Kesäniemi*) Колін Ватсон (*Colin Watson*) Франсуа-Ерік Гуйомарх (*François-Eric Guyomarc'h*) Крістінель Думітру (*Cristinel Dumitru*) Джеймс Холланд (*James Holland*) Гарі Робінсон (*Gary Robinson*) Стівен де Вріес (*Stephen de Vries*) Гленн Тен Кейт (*Glenn Ten Cate*) Ріккардо Тен Кейт (*Riccardo Ten Cate*) Мартін Кноблох (*Martin Knobloch*) Абхінав Седжпал (*Abhinav Sejpal*) Девід Раян (*David Ryan*) Стівен ван дер Баан (*Steven van der Baan*) Раян Девурст (*Ryan Dewhurst*) Рауль Ендрес (*Raoul Endres*) Роберто Мартеллоні (*Roberto Martelloni*)

Версія 2.0, 2014 р.

Керівники проекту	Провідні автори	Співавтори та рецензенти
Даніель Кусберт (*Daniel Cuthbert*) Сахба Казерооні (*Sahba Kazerooni*)	Ендрю ван дер Сток (*Andrew van der Stock*) Крішна Раджа (*Krishna Raja*)	Антоніо Фонтес (*Antonio Fontes*) Колін Ватсон (*Colin Watson*) Джеф Серджент (*Jeff Sergeant*) Пекка Сілланпаа (*Pekka Sillanpää*) Архангел Куісон (*Archangel Cuison*) Д-р Емін Татлі (*Dr. Emin Tatli*) Жером Атіас (*Jerome Athias*) Сафуат Хамді (*Safuat Hamdy*) Арі Кезеніемі (*Ari Kesäniemi*) Етьєн Сталманс (*Etienne Stalmans*) Джим Маніко (*Jim Manico*) Скотт Люк (*Scott Luc*) Бой Баукема (*Boy Baukema*) Еван Гаустад (*Evan Gaustad*) Маіт Пеекма (*Mait Peekma*) Себастьян Делеерснідер (*Sebastien Deleersnyder*)

Версія 1.0, 2009 р.

Керівники проекту	Провідні автори	Співавтори та рецензенти
Майк Боберскі (*Mike Boberski*) Джеф Вільямс (*Jeff Williams*) Дейв Вічерс (*Dave Wichers*)		Ендрю ван дер Сток (*Andrew van der Stock*) Д-р Сарбарі Гупта (*Dr. Sarbari Gupta*) Джон Стівен (*John Steven*) П'єр Парренд (*Pierre Parrend*) Беррі Бойд (*Barry Boyd*) Д-р Томас Браун (*Dr. Thomas Braun*) Кен Хуанг (*Ken Huang*) Річард Кемпбелл (*Richard Campbell*) Бедірхан Ургун (*Bedirhan Urgun*) Еоін Кеарі (*Eoin Keary*) Кетан Діліпкумар В'яс (*Ketan Dilipkumar Vyas*) Скотт Мацумото (*Scott Matsumoto*) Колін Ватсон (*Colin Watson*) Гауранг Шах (*Gaurang Shah*) Ліз Фонг (*Liz Fong*) Шоувік Бардхан (*Shouvik Bardhan*) Ден Корнелл (*Dan Cornell*) Джордж Лоулес (*George Lawless*) Мендіп Кера (*Mandeep Khera*) Стен Віссеман (*Stan Wisseman*) Дейв Хаусладен (*Dave Hausladen*) Джефф ЛоСапіо (*Jeff LoSapio*) Метт Прессон (*Matt Presson*) Стівен де Вріес (*Stephen de Vries*) Теодор Віноград (*Theodore Winograd*) Джеремая Гроссман (*Jeremiah Grossman*) Нем Нгуєн (*Nam Nguyen*) Стів Койл (*Steve Coyle*) Дейв ван Штайн (*Dave van Stein*) Джон Мартін (*John Martin*) Пол Доутхіт (*Paul Douthit*) Террі Діаз (*Terrie Diaz*)

Про *Стандарт*

Стандарт оцінювання відповідності безпеки додатків є списком вимог до безпеки додатків, а також її тестування, яким можуть користуватися архітектори, розробники, тестувальники, фахівці в галузі безпеки та звичайні споживачі для визначення того, чи додаток є безпечним.

Авторські права та ліцензія

Передмова

Ласкаво просимо до *ASVS*[1] версії 3.0. *Стандарт* розроблено групою фахівців з метою створення основних положень, які окреслюють вимоги щодо безпеки та її контролі. *ASVS* зосереджується на стандартизації функціональних і нефункціональних контролів безпеки, необхідних при проектуванні, розробці та тестуванні сучасних веб-додатків.

ASVS версії 3.0 є поєднанням спільних зусиль його розробників і відгуків галузевих фахівців. Важливим у цьому виданні є опис досвіду практичного впровадження *ASVS*. Він може як допомогти тим, хто вперше зустрівся зі *Стандартом*, у плануванні його впровадження, так і познайомити компанії, які вже працювали з ним, із досвідом інших користувачів.

Завжди існує велика ймовірність того, що зі *Стандартом* можуть не погоджуватися на всі 100 %. Аналіз ризиків є певною мірою суб'єктивним, що створює виклик спробам узагальнити все в універсальному стандарті. Проте існує надія, що останні оновлення, представлені в цій версії, є кроком у правильному напрямку та підсилюють концепції, окреслені в цьому важливому галузевому стандарті.

Що нового у версії 3.0?

Версія 3.0 була доповнена декількома новими розділами, у тому числі про конфігурації, веб-служби та сучасні клієнтські додатки, щоб зробити *Стандарт* кориснішим для сучасних і, як правило, гнучких додатків із розповсюдженим користувацьким інтерфейсом або мобільним клієнтом на основі *HTML5*[2], що звертаються до загального комплексу веб-служб, побудованих за архітектурою *REST*[3], які використовують *SAML*[4]-автентифікацію.

Окрім того, *Стандарт* було дедубліковано, у тому числі з метою виключення багаторазового тестування одного і того самого мобільного додатка розробником.

У версії 3.0 подається посилання на словник *CWE*[5], який може допомогти при визначенні такої інформації, як ймовірність використання та результати успішного використання, а також у широкому сенсі для розуміння того, що може піти не так, якщо контролі безпеки не застосовуються або застосовуються неефективно, та як нівелювати слабкі сторони.

Зрештою, було налагоджено діалог із громадськістю та проведено сесії для розгляду *Стандарту* фахівцями в рамках конференції з безпеки додатків *AppSec EU 2015* у Європі та заключну робочу сесію в рамках конференції з безпеки додатків *AppSec USA 2015* у США, щоб врахувати у *Стандарті* відгуки багатьох фахівців. Під час розгляду *Стандарту* фахівцями було істотно

[1] ASVS (Application Security Verification Standard) – Стандарт оцінювання відповідності безпеки додатків.
[2] HTML5 (HyperText Markup Language 5) – мова розмітки гіпертекстових документів, версія 5.
[3] REST (Representational State Transfer) – передача репрезентативного стану.
[4] SAML (Security Assertion Markup Language) – мова розмітки декларації безпеки.
[5] CWE (Common Weakness Enumeration) – Перелік розповсюджених слабких сторін.

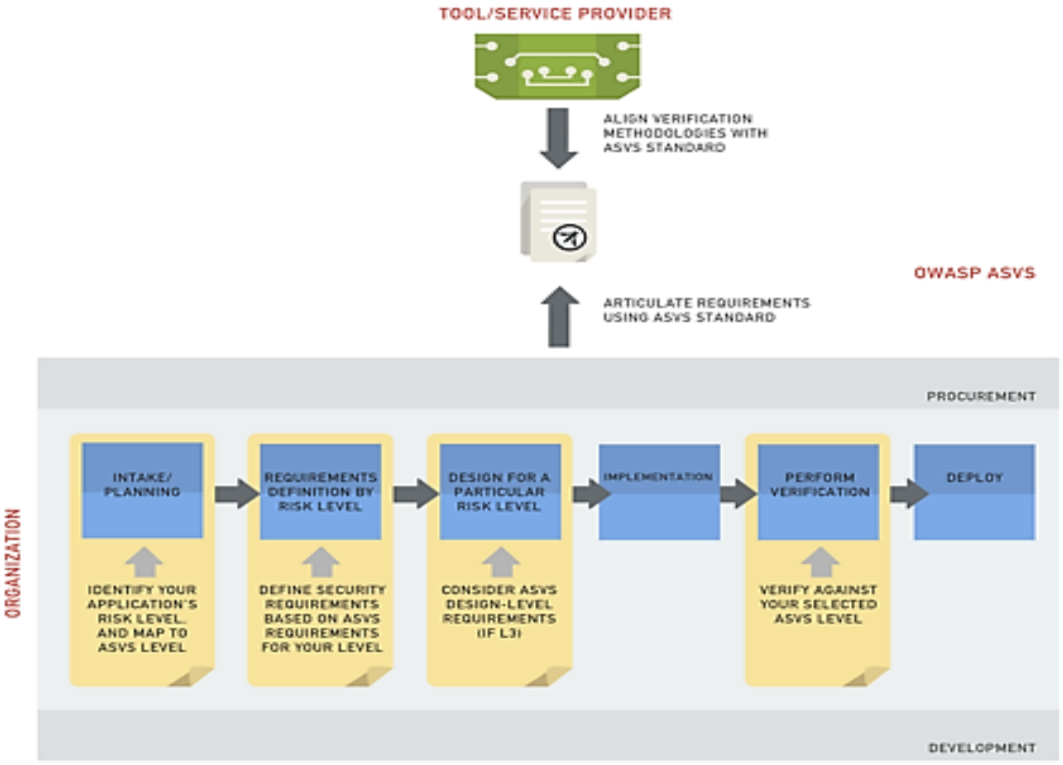
змінено значення одного з контролів внаслідок створення нового контролю та відмови від відповідного старого. У *Стандарті* свідомо не описуються нерекомендовані вимоги до контролю, оскільки це могло би спричинити плутанину. Вичерпний опис змін наведено у Додатку А.

Загалом, версія 3.0 є єдиною та найбільшою зміною *Стандарту* за час його існування. Розробники *Стандарту* сподіваються що він буде корисним, а також що користувачі застосовуватимуть його найрізноманітнішими способами.

Застосування *Стандарту оцінювання відповідності безпеки додатків*

Існує дві головні цілі *ASVS*:

1) допомогти організаціям у розробці та обслуговуванні захищених додатків;
2) забезпечити узгодження вимог і пропозицій служб безпеки, постачальників інструментів забезпечення захисту та споживачів.



Малюнок 1. Застосування *ASVS* організаціями та постачальниками інструментів / послуг.

Рівні оцінювання відповідності безпеки додатків

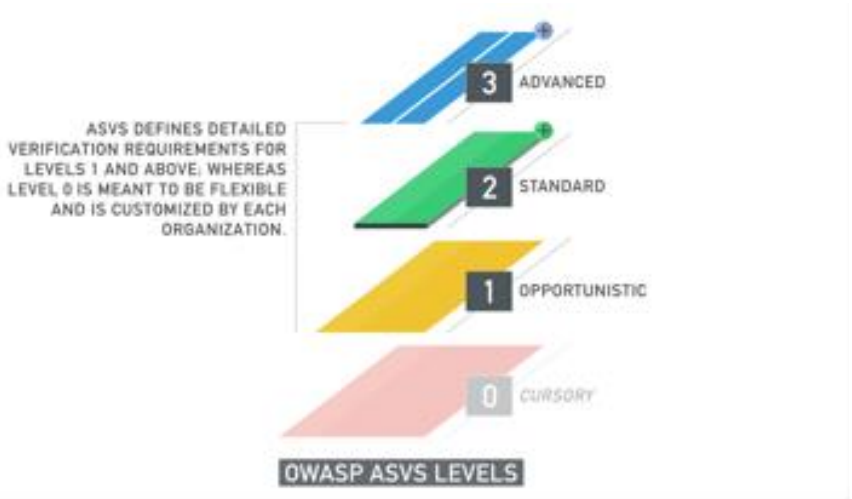
Стандарт оцінювання відповідності безпеки додатків визначає три рівні оцінювання відповідності, кожен із яких є глибшим від попереднього.

Рівень 1 *ASVS* призначений для всіх програмних засобів.

Рівень 2 *ASVS* призначений для додатків, що містять конфіденційні дані, які потребують захисту.

Рівень 3 *ASVS* призначений для найбільш критичних додатків, за допомогою яких виконуються фінансові операції, що містять конфіденційні медичні дані або вимагають найвищого рівня довіри до себе.

Кожен із рівнів *ASVS* має список вимог до безпеки. Такі вимоги, окрім іншого, можна віднести до основних характеристик і функціональних можливостей безпеки, які розробники повинні вбудовувати у свої програмні засоби.



Малюнок 2. Рівні *ASVS OWASP*.

Як застосовувати *Стандарт*?

Одним із найкращих способів застосування *Стандарту оцінювання відповідності безпеки додатків* є його використання як макету для створення контрольного списку безпечного кодування для конкретного додатка, платформи чи організації. Застосування *ASVS* у різноманітних практичних сценаріях дозволить детальніше зосередитися на тих вимогах до безпеки, які є найбільш важливими для конкретного проекту та середовища.

Рівень 1: поверхневий

Додаток досягає першого (або поверхневого) рівня *ASVS*, якщо він належним чином захищений від вразливостей, які легко виявляються та входять у перелік веб-вразливостей *Топ-10 OWASP* та інші схожі списки.

Рівень 1, як правило, підходить для додатків, які вимагають низького рівня довіри до правильності використання контролів безпеки, або для забезпечення швидкого аналізу організацією групи додатків, або для допомоги при розробці переліку пріоритезованих вимог до безпеки в рамках виконання багатофазової роботи. Контролі першого рівня можуть забезпечуватися автоматично як інструментами, так і просто вручну без доступу до вихідного коду. Рівень 1 вважається мінімальним для всіх додатків.

Найбільш ймовірно, що такі додатки поставатимуть перед загрозами від зломщиків, які застосовують прості малоефективні техніки для визначення вразливостей, що легко виявити та використати, на противагу цілеспрямованим зломщикам, які фокусують свою енергію на певному цільовому додатку. Якщо дані, що обробляються додатком, мають високу цінність, першого рівня часто буде недостатньо.

Рівень 2: стандартний

Додаток досягає другого (або стандартного) рівня *ASVS*, якщо він належним чином захищений від більшості ризиків сьогодення, пов’язаних із програмними засобами.

Рівень 2 забезпечує наявність ефективних контролів безпеки, які застосовуються в самому додатку. Як правило, він підходить для додатків, які проводять важливі міжкорпоративні фінансові операції, обробляють медичну інформацію, виконують критично важливі для бізнесу або конфіденційні функції або обробляють інші конфіденційні активи.

Як правило, такі додатки поставатимуть перед загрозами від досвідчених мотивованих зломщиків, які зосереджуються на конкретних цілях і працюють за допомогою інструментів і технік, що часто та ефективно застосовуються при виявленні та використанні слабких сторін додатка.

Рівень 3: ґрунтовний

Третій рівень є найвищим рівнем оцінювання відповідності в рамках *ASVS*. Як правило, цей рівень необхідний для додатків, які вимагають оцінювання відповідності безпеки на глибоких рівнях, наприклад, у військовій галузі, у сфері охорони здоров’я та безпеки, в критичній інфраструктурі тощо. Рівень 3 *ASVS* може бути необхідним для організацій у тому випадку, якщо порушення безпеки додатка може мати значний вплив на діяльність організації, чи навіть на її життєздатність. Нижче наведено приклад вказівок із застосування третього рівня *ASVS*. Додаток досягає третього (або ґрунтовного) рівня *ASVS*, якщо він належним чином захищений від спрямованих вразливостей і демонструє принципи надійної системи захисту.

Додаток, що відповідає третьому рівню *ASVS*, вимагає більш глибокого аналізу, архітектури, кодування та тестування, ніж додатки інших рівнів. Захищений додаток має серйозну модульну організацію (для посилення, наприклад, стійкості, масштабованості та, перш за все, рівнів безпеки). Кожен модуль (відокремлений мережевим з'єднанням та / або фізичним екземпляром) має власні функціональні обов'язки щодо безпеки (захист у глибину), які необхідно належним чином документувати. Ці обов'язки охоплюють контролі забезпечення конфіденційності (наприклад, шифрування), цілісності (наприклад, перевірки вхідних даних), доступності (наприклад, коректного управління навантаженнями), автентифікації (у тому числі між системами), невідмовності від дії, авторизації та аудиту (реєстрування).

Практичне застосування *ASVS*

Мотивації загроз різняться. Деякі галузі мають унікальні інформаційні та технологічні активи та нормативні вимоги до відповідності, що залежать від галузі.

Нижче наведено вказівки щодо рекомендованих рівнів *ASVS*. Хоча існують певні унікальні критерії та відмінності загроз для кожної індустрії червоною ниткою по всіх галузевих сегментах проходить твердження про те, що випадкові зломщики завжди шукають легковразливі додатки. Саме тому незалежно від галузі всі додатки повинні відповідати першому рівню *ASVS*. Це – рекомендована відправна точка управління ризиками, які найлегше виявити. Організаціям наполегливо рекомендується звернути більшу увагу на характеристику їх унікальних ризиків залежно від сфери їх діяльності. З іншого боку стоїть третій рівень *ASVS*, який може поставити під загрозу безпеку людини або серйозно вплинути на організацію у випадку цілковитого порушення безпеки додатка.

Галузь	Профіль загрози	Рекомендується P1 *ASVS*	Рекомендується P2 *ASVS*	Рекомендується P3 *ASVS*
Фінанси та страхування	Хоча цей сегмент пробують атакувати і випадкові зломщики, частіше він є пріоритетною мішенню для мотивованих хакерів. Відповідно, атаки часто мають фінансове підгрунтя . Зазвичай зломщиків цікавить конфіденційна інформація або дані облікового запису, які можна використати з метою шахрайства або для отримання безпосередньої вигоди від використання вбудованих у додатки функціональних можливостей,	Усі додатки, що мають доступ до мережі.	Додатки, що містять конфіденційну інформацію, як, наприклад, номер кредитної картки чи особисті дані, за допомогою якої можна переводити обмежені суми грошей обмеженими способами, наприклад: (i) переводити гроші між рахунками однієї установи; (ii) здійснювати повільний грошовий обіг (наприклад, через *ACH*[7]) із обмеженнями на транзакції; (iii) здійснювати безготівкові грошові перекази зі жорсткими обмеженнями на перекази коштів протягом певного проміжку часу.	Додатки, що містять велику кількість конфіденційної інформації або дозволяють швидкий переказ великих сум грошей (наприклад, безготівкові грошові перекази) та / або переведення великих сум грошей за допомогою окремих операцій або низки менших переказів.

[7] ACH (Automated Clearing House) – Автоматична розрахункова палата.

	пов'язаних із грошовим обігом. З-поміж технік можна виділити крадіжки облікових даних, атаки на рівні додатків і соціальну інженерію. Деякі з основних документів про відповідність: *PCI DSS*[6], Закон Гремма-Ліча-Блайлі та Закон Сарбейнса-Окслі.			
Виробництво, професійна діяльність, транспортування, технології, комунальні послуги, інфраструктура та оборона	Хоча може здатися, що ці галузі мають дуже мало спільного, проте зломщики, які найбільш ймовірно атакуватимуть організації цих сегментів, найчастіше здійснюють цілеспрямовані атаки, затрачаючи на них більше часу, зусиль та ресурсів. Виявлення конфіденційної інформації та проникнення в захищені системи часто є складним завданням, яке вимагає залучення інсайдерів і застосування технік, які ґрунтуються на індивідуальній психології. Такі атаки можуть здійснюватися інсайдерами, сторонніми особами або їх об'єднаними зусиллями. Їх мета може полягати в отриманні доступу до інтелектуальної власності для стратегічної чи технологічної вигоди. Окрім того, варто звернути увагу на зломщиків, метою яких зловживання функціональними можливостями додатків, вплив на поведінку систем, що містять конфіденційну інформацію, або	Усі додатки, що мають доступ до мережі.	Додатки, що містять інформацію для внутрішнього використання або інформацію про співробітників, яку можна застосувати в соціальній інженерії. Додатки, що містять несуттєві дані, які все ж відносяться до важливої інтелектуальної власності або комерційних таємниць.	Додатки, що містять цінну інтелектуальну власність або комерційні чи державні таємниці (наприклад, у США це – інформація, яка класифікується як секретна або вище), що є критично важливими для подальшого існування організації або її успішної діяльності. Додатки, які контролюють функціональні можливості, пов'язані з конфіденційними даними (наприклад, транзит, виробниче обладнання або системи управління), або які можуть загрожувати безпеці життя.

[6] PCI DSS (Payment Card Industry Data Security Standard) – Стандарт безпеки даних індустрії банківських платіжних карток.

	порушення їх роботи. Більшість зломщиків намагаються отримати конфіденційну інформацію, яку можна застосувати з метою отримання прямої чи опосередкованої вигоди шляхом використання ідентифікаційних і платіжних даних. Часто такі дані використовуються для крадіжки особистості, проведення шахрайських платежів або в різноманітних шахрайських схемах.			
Охорона здоров'я	Більшість зломщиків намагаються отримати конфіденційну інформацію, яку можна застосувати з метою отримання прямої чи опосередкованої вигоди шляхом використання ідентифікаційних і платіжних даних. Часто такі дані використовуються для крадіжки особистості, проведення шахрайських платежів або в різноманітних шахрайських схемах. У галузі охорони здоров'я США це – правила постанови *HIPAA*[8] щодо конфіденційності, безпеки та сповіщення про їх порушення, а також щодо безпеки пацієнтів (http://www.hhs.gov/ocr/privacy/).	Усі додатки, що мають доступ до мережі.	Додатки, що містять невелику чи помірну кількість конфіденційної медичної інформації (закриту медичну інформацію), ідентифікаційні або платіжні дані.	Додатки, що контролюють медичне обладнання та прилади або записи, які можуть загрожувати життю людини. Платіжні системи та системи розрахункових терміналів, що містять значну кількість даних про транзакції, які можна використати з метою шахрайства, включаючи інтерфейси адміністратора таких додатків.
Роздрібна торгівля, харчування, готельно-ресторанна справа	У цьому сегменті багато зломщиків застосовують тактику випадкових розбійних нападів. Однак, все ж існує постійна загроза спрямованих атак на додатки, які містять платіжну інформацію, проводять фінансові операції або зберігають інформацію, що дозволяє ідентифікувати особистість. Окрім того,	Усі додатки, що мають доступ до мережі.	Бізнес-додатки, інформація щодо каталогів продукції, внутрішньокорпоративна інформація та додатки з обмеженою інформацією про користувачів (наприклад, з контактною інформацією). Додатки, що містять невелику чи помірну кількість платіжних даних або функціональні можливості, пов'язані з оформленням та оплатою замовлень.	Платіжні системи та системи розрахункових терміналів, що містять значну кількість даних про транзакції, які можна використати з метою шахрайства, включаючи інтерфейси адміністратора таких додатків. Додатки з великими обсягами конфіденційної інформації, як, наприклад, повні номери кредитних карток, дівоче прізвище матері, номери соціального страхування тощо.

[8] HIPAA (Health Insurance Portability and Accountability Act) – Закон про безперервність дії та прозорість медичного страхування.

	в цьому галузевому сегменті існує менш ймовірна загроза складніших атак з метою крадіжки інтелектуальної власності, конкурентної розвідки або отримання переваги над цільовою організацією або діловим партнером у переговорах.			

Практичні приклади

Практичний приклад 1. *ASVS* як настанови щодо тестування безпеки

В одному приватному університеті Юти, США, місцева «Червона команда» застосовує *ASVS OWASP* як настанови при проведенні тестів на проникнення. *Стандарт* використовується протягом усього процесу тестування: на перших зборах по плануванню та визначенню обсягу тестування, в самому процесі тестування як настанови, а також при оформленні результатів тестування в заключному звіті для клієнтів. Окрім того, «червоні» проводять навчання групи, яка застосовує *ASVS*.

«Червона команда» проводить тести на проникнення на рівні мереж і додатків для різних факультетів ВНЗ в рамках всеуніверситетської стратегії інформаційної безпеки. На перших зборах по плануванню клієнти часто не поспішають давати згоду на тестування їх додатків групою студентів. Після ознайомлення зацікавлених сторін із *ASVS* і пояснення їм того, що тестування проводитиметься згідно зі *Стандартом*, а також того, що заключний звіт міститиме опис відповідності додатка *Стандарту*, занепокоєння негайно зникає. Потім *ASVS* застосовується при визначенні обсягу тестування, щоб встановити, скільки необхідно часу та зусиль для проведення тестування. «Червона команда» пояснює ризик-орієнтоване тестування на основі попередньо визначених рівнів оцінювання відповідності *ASVS*. Це сприяє досягненню взаєморозуміння щодо необхідного обсягу тестування певного додатка між клієнтом, зацікавленими сторонами та групою.

Як тільки починається тестування, «червоні» застосовують *ASVS* для організації своєї діяльності та розподілу робочого навантаження. Керівники проекту можуть легко контролювати прогрес тестування додатка групою, відслідковуючи, які вимоги до оцінювання відповідності вже перевірені, а які ще підлягають перевірці, що у свою чергу покращує комунікацію з клієнтами та дозволяє керівникам проекту ефективніше управляти ресурсами. Оскільки «Червона команда» головним чином складається зі студенів, перед більшістю членів групи стоїть чимало вимог, пов’язаних із різними курсами, які займають багато часу. Чітко визначені завдання, які ґрунтуються на цілих категоріях або окремих вимогах до оцінювання відповідності, допомагають членам групи розуміти, що саме необхідно тестувати, а також точно визначати час виконання таких завдань. Окрім того, чітка структура *ASVS* корисна і при звітуванні, оскільки члени групи можуть фіксувати результати до переходу на наступне завдання, ефективно оформляючи більшість звітів одночасно з проведенням тестів на проникнення.

Заключний звіт «Червоної команди» ґрунтується на *ASVS*, описуючи статус кожної вимоги до оцінювання відповідності і, за необхідності, надаючи додаткову інформацію. Це чітко показує клієнтам і зацікавленим особам, де знаходиться їх додаток згідно зі *Стандартом*, а також є надзвичайно важливим при визначенні подальших завдань, оскільки демонструє підвищення чи зниження рівня безпеки протягом тривалого періоду. Більше того, зацікавлені сторони можуть легко знайти інформацію про те, як додаток виконав певну категорію чи категорії, оскільки формат

звіту повністю співпадає з *ASVS*. Окрім того, чітка структура *ASVS* полегшує і навчання нових членів групи тому, як створювати звіти на основі форматів минулих звітів.

Зрештою, після впровадження *ASVS* покращився і процес навчання «Червоної команди». Раніше щотижневі навчання зосереджувалися на темі, яку обирали керівники проекти або групи по запиту членів групи чи залежно від їх потреб. Навчання, які ґрунтувалися на цих критеріях, покращували навички членів групи, але не завжди були пов'язані з основною діяльністю «червоних». Іншими словами, у тестуваннях на проникнення група значно кращою не ставала. Тепер, після впровадження *ASVS*, навчання групи зосереджуються на окремих вимогах до оцінювання відповідності, що призвело до суттєвого покращення навичок окремих членів групи та якості заключного звіту.

Практичний приклад 2. *ASVS* як захищений життєвий цикл програмного забезпечення

Стартап, який займається наданням послуг із аналітики великих даних фінансовим установам, усвідомлює, що безпека в процесі розробки програмного забезпечення є найважливішою вимогою для отримання доступу до фінансових метаданих та їх обробки. Саме тому цей стартап обрав *ASVS* за основу безпеки в гнучкому життєвому циклі розробки.

Стартап застосовує *ASVS* при створенні збірок користувацьких історій та варіантів використання для вирішення питань, пов'язаних із функціональною безпекою, як, наприклад, при визначенні найкращого способу впровадження функціональної можливості входу в систему. Стартап застосовує *ASVS* не так, як більшість інших користувачів: у *Стандарті* він підбирає ті вимоги, які підходять йому в конкретному спринті, та додає їх безпосередньо до переліку функціональних вимог, якщо це – функціональна можливість, або до обмежень діючих сценаріїв, якщо це – нефункціональна можливість. Наприклад, при виборі двохфакторної *TOTP*[9]-автентифікації було додано політику паролів і регулятор веб-служб, що одночасно стало механізмом виявлення та попередження атак методом перебору. У наступних спринтах буде вибрано додаткові вимоги за схемами «саме вчасно» (*just in time)* та «Вам це ніколи не знадобиться» (*you ain't gonna need it*).

Розробники застосовують *ASVS* як контрольний перелік для взаємної перевірки, що запобігає небезпечним практикам кодування, а також у ретроспективних планах для того, щоб розробники, які розробили нову функціональність, могли перевірити всі ймовірні вимоги *ASVS* і те, чи необхідно щось покращувати чи спрощувати у наступних спринтах.

Зрештою, розробники застосовують *ASVS* в рамках захищеного блоку автоматизованого оцінювання відповідності та набору інтеграційних тестів для перевірки варіантів використання, неправильного використання та фаззингу. Мета полягає в тому, щоб знизити ризик водоспадного методу, в якому тестування на проникнення виконується наприкінці розробки додатку, що може

[9] TOTP (Time-based One-time Password) – часовий одноразовий пароль.

призвести до дорогої реорганізації коду, коли поправки вносяться до вже працюючого продукту. Оскільки після кожного спринта можуть з‘являтися нові версії додатка, недостатньо покладатися на одноразову перевірку. Таким чином, при автоматизації режиму тестування на проникнення не повинно залишитися жодної значної проблеми, яку міг би виявити кваліфікований контролер безпеки після багатотижневої перевірки додатка.

Програмні засоби для оцінки відповідності

Позиція *OWASP* щодо сертифікатів і знаків довіри *ASVS*

OWASP як комерційно нейтральна неприбуткова організація не видає жодні сертифікати постачальникам послуг, контролерам відповідності чи на програмне забезпечення.

OWASP офіційно не перевіряв, не реєстрував і не видавав жодні твердження, знаки довіри та сертифікати, тому організаціям, які покладаються на них, варто бути обережними, довіряючи будь-яким третім сторонам або знакам довіри, що стверджують про наявність сертифікату *ASVS*.

Однак, організаціям не забороняється пропонувати такі послуги з підтвердження відповідності, якщо вони не стверджують про наявність офіційних сертифікатів *OWASP*.

Настанови сертифікуючим організаціям

Стандарт оцінювання відповідності безпеки додатків можна застосовувати як «відкриту книгу» з оцінювання відповідності додатка, включаючи відкритий необмежений доступ до його основних ресурсів, як, наприклад, до архітектури і розробників, проектної документації, вихідного коду, автентифікованого доступу до систем тестування (в тому числі доступу до щонайменше одного облікового запису в кожній ролі), зокрема для оцінювання відповідності P2 та P3.

Історично тестування на проникнення та аналіз безпеки програмного коду містили лише проблемні знахідки аудиту. Тобто, в заключному звіті описувалася виключно інформація про невдачі. Сертифікуючі організації повинні включати в кожен звіт обсяг оцінювання відповідності (зокрема, якщо основний компонент знаходиться поза діапазоном, як, наприклад, *SSO*[10]-автентифікація) та стислий виклад результатів оцінювання відповідності, в тому числі про пройдені та непройдені тести з чіткими вказівками, як вирішити питання з непройденими тестами.

Звичайною галузевою практикою є ведення детальної робочої документації, збереження знімків екрану або відео, сценаріїв для надійного повторного використання та електронних записів тестування, як, наприклад, перехоплення журналів проксі-сервера та відповідних заміток, таких як список очистки. Така інформація може бути дійсно корисною для доказу результатів розробникам, які мають найбільші сумніви. Недостатньо просто запустити інструмент – і відзвітувати про невдачу. Це (в жодному разі) не надає достатньої кількості доказів про ретельне тестування всіх аспектів на рівні сертифікації. У випадку виникнення спірних питань необхідно, щоб було достатньо доказів для надання впевненості в тому, що було протестовано кожну без виключення вимогу, пов’язану з оцінюванням відповідності.

[10] SSO (Single Sign-On) – технологія єдиного входу.

Роль інструментів автоматизованого тестування на проникнення

Бажано, щоб інструменти автоматизованого тестування на проникнення забезпечували максимально можливе покриття та задіювали якомога більше параметрів безліччю різних форм зловмисних введень.

Оцінювання відповідності *ASVS* неможливо завершити повністю лише за допомогою інструментів автоматизованого тестування на проникнення. Хоча значну кількість вимог P1 можна перевірити за допомогою автоматизованого тестування, абсолютна більшість вимог не піддається перевірці інструментами автоматизованого тестування на проникнення.

Варто зауважити, що з розвитком галузі безпеки додатків межа між автоматизованим і ручним тестуванням все більше розмивається. Інструменти автоматизованого тестування часто вимагають ручного налаштування фахівцями, а контролери безпеки, які працюють вручну, часто використовують різноманітні інструменти автоматизованого тестування.

Роль тестування на проникнення

Цілком можливо виконати тестування на проникнення та перевірити усі аспекти на відповідність P1 вручну без доступу до вихідного коду, але це не є провідною практикою. Для тестування P2 необхідно щонайменше отримати доступ до розробників, документації та коду, а також автентифікований доступ до системи. P3 неможливо повністю охопити тестуванням на проникнення, оскільки більшість додаткових завдань відноситься до оцінювання системних конфігурацій, аналізу шкідливих кодів, моделювання загроз та інших робочих продуктів, не пов'язаних із тестуванням на проникнення.

ASVS як детальні настанови щодо архітектури безпеки

Одним із найрозповсюдженіших є застосування *Стандарту оцінювання відповідності безпеки додатків* як ресурсу архітекторів безпеки. Двом основним методологіям архітектури безпеки – *SABSA* та *TOGAF* – бракує великої кількості інформації, необхідної для аналізу архітектури безпеки додатка. *ASVS* можна застосовувати для заповнення цих прогалин, оскільки він допомагає архітекторам безпеки обирати кращі контролі для типових проблем, як, наприклад, шаблони захисту даних і стратегії перевірки вхідних даних.

ASVS замість стандартів безпечного кодування

Багато організацій можуть почерпнути користь із впровадження *ASVS*, обравши один із трьох рівнів або розгалузивши *Стандарт* та внісши всі зміни, необхідні для обраного рівня ризику додатка, залежно від галузі застосування. Рекомендується саме такий вид розгалуження, оскільки при ньому підтримується відстеження. Таким чином, якщо додаток пройшов перевірку та

відповідає вимозі 4.1, це означає те саме і для його розгалужених копій, які розвиваються згідно зі *Стандартом*.

ASVS як настанови щодо модульних і комплексних автоматизованих тестувань

ASVS є високо придатним для тестування, за єдиним винятком, пов'язаним із вимогами до архітектури і шкідливих кодів. Завдяки модульним і комплексним тестам, які перевіряють на окремі та характерні варіанти неправильного використання та фаззингу, з кожною новою версією додаток може практично здійснювати самооцінювання. Наприклад, можна доповнити додатковими тестами комплект тестів управління входом в систему, щоб перевіряти параметри імені користувача на типовість, вести список облікових записів, попереджати атаки методом перебору, а також вставлення в запити *LDAP*[11] та *SQL*[12]*, XSS*[13] *тощо*. Подібним чином, тестування параметрів паролю повинно охоплювати типові паролі, довжину паролю, ін'єкцію нульового байта, видалення параметру, *XSS*, перелік облікових записів, та ін..

ASVS як навчання щодо безпеки розробки програм

Окрім того, *ASVS* можна застосовувати для визначення характерних особливостей захищених програмних засобів. Багато курсів «безпечного кодування» є просто курсами етичного хакерства, де дають надзвичайно мало порад щодо кодування. Це ніяк не допомагає розробникам. Натомість, курси безпечної розробки можуть застосовувати *ASVS*, чітко зосереджуючись на проактивних контролях, описаних у *Стандарті*, а не на топ-десятці негативних дій, яких слід уникати.

[11] LDAP (Lightweight Directory Access Protocol) – полегшений протокол доступу до директорій.
[12] SQL (Structured Query Language) – мова структурованих запитів.
[13] XSS (Cross-Site Scripting) – міжсайтове виконання сценаріїв.

Проекти *OWASP*, що застосовують *ASVS*

Security Knowledge Framework

https://www.owasp.org/index.php/OWASP_Security_Knowledge_Framework

SKF[14] застосовується для навчання розробників написанню безпечного коду. Це – веб-додаток на основі відкритого вихідного коду *Python*[15]-*Flask*[16], який застосовує *Стандарт оцінювання відповідності безпеки додатків OWASP* у навчанні написанню безпечного за дизайном коду.

Zed Attack Proxy OWASP

https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project

ZAP[17] *OWASP* є простим у використанні інструментом комплексного тестування на проникнення для виявлення вразливостей у веб-додатках, розроблений для людей із найрізноманітнішим досвідом у сфері безпеки. Сам по собі цей інструмент є ідеальним для розробників і функціональних тестувальників, які є новачками в тестуваннях на проникнення. *ZAP* надає автоматизовані сканери, а також набір інструментів, які дозволяють вручну знаходити вразливості системи безпеки.

Cornucopia OWASP

https://www.owasp.org/index.php/OWASP_Cornucopia

Cornucopia OWASP – це карткова гра, за допомогою якої групи розробників програмного забезпечення визначають вимоги до безпеки у процесі гнучких, традиційних і формальних розробок. Вона не залежить від мови, платформи та технологій. (Набори *Cornucopia* були обрані згідно зі структурою *Короткого довідника SCP*[18] *OWASP* із розглядом окремих розділів у *Стандартах оцінювання відповідності безпеки додатків OWASP*, у *Настановах щодо тестування OWASP* і в *Принципах безпечних розробок* Девіда Рука.

[14] SKF (Security Knowledge Framework) – Основи знань з безпеки.
[15] Python – інтерпретована об'єктно-орієнтована мова програмування високого рівня з динамічною семантикою.
[16] Flask – мікрокаркас для створення веб-додатків мовою програмування Python.
[17] ZAP (Zed Attack Proxy) – сканер безпеки веб-додатків з відкритим кодом Проксі зед-атак.
[18] SCP (Secure Coding Practices) – Практики захисного кодування.

Детальні вимоги до оцінювання відповідності

ОВ1. Архітектура, проектування та моделювання загроз

ОВ2. Автентифікація

ОВ3. Управління сеансами

ОВ4. Контроль доступу

ОВ5. Обробка шкідливих вхідних даних

ОВ7. Криптографія при зберіганні

ОВ8. Обробка та реєстрація помилок

ОВ9. Захист даних

ОВ10. Комунікації

ОВ11. Конфігурації безпеки *HTTP*

ОВ13. Контроль за шкідливими програмними засобами

ОВ15. Бізнес-логіка

ОВ16. Файли та ресурси

ОВ17. Мобільні додатки

ОВ18. Веб-служби (НОВЕ у 3.0)

ОВ19. Конфігурації (НОВЕ у 3.0)

ОВ1. Архітектура, проектування та моделювання загроз

Задача

Забезпечення відповідності оцінюваного додатка наступним високорівневим вимогам:

- на першому рівні компоненти додатка визначаються та входять до складу додатка небезпричинно;
- на другому рівні визначається архітектура, а код відповідає її вимогам;
- на третьому рівні архітектура та дизайн наявні, використовуються та є ефективними.

Примітка. Цей розділ повернули у версію 3.0, залишивши, по суті, такі самі контролі архітектури, як у *ASVS* версії 1.0.

Вимоги

№	Опис	1	2	3	З версії
1.1	Перевірити, чи визначені всі компоненти додатка, і чи вони потрібні.	✓	✓	✓	1.0
1.2	Перевірити, чи визначені всі компоненти, як, наприклад, бібліотеки, модулі та зовнішні системи, які не є частиною додатка, але необхідні для його функціонування.		✓	✓	1.0
1.3	Перевірити, чи було визначено високорівневу архітектуру додатка.		✓	✓	1.0
1.4	Перевірити, чи всі компоненти додатка витікають з урахування бізнес-функцій та / або функцій безпеки, які вони виконують.			✓	1.0
1.5	Перевірити, чи всі компоненти, які не є частиною додатка, але необхідні для його функціонування, витікають з урахування бізнес-функцій та / або функцій безпеки, які вони виконують.			✓	1.0
1.6	Перевірити наявність моделі загроз для цільового додатка, а також те, чи така модель охоплює всі ризики, пов'язані зі *STRIDE*[19].			✓	1.0
1.7	Перевірити наявність централізованого впровадження всіх контролів безпеки (включаючи бібліотеки, які викликають зовнішні служби безпеки).			✓	1.0
1.8	Перевірити, чи відділені компоненти один від одного за допомогою певного контролю безпеки, як, наприклад, мережевою сегментацією, правилами міжмережевого екрану або групами хмарної безпеки.		✓	✓	3.0
1.9	Перевірити, чи додаток має чіткий розподіл між рівнем даних, рівнем контролера та рівнем відображення, щоб рішення щодо безпеки могли прийматися в захищених компонентах.		✓	✓	3.0
1.10	Перевірити, чи клієнтський код не містить конфіденційної бізнес-логіки, секретних ключів або іншої конфіденційної інформації.		✓	✓	3.0

[19] STRIDE (Spoofing, Tampering, Repudiation, Information Disclosure, and Elevation) – система класифікації загроз підміни ідентифікатора користувача, втручання, відмови, розголошення даних, відмови в обслуговуванні та підвищення привілеїв.

Посилання

Детальнішу інформацію наведено в наступних документах:

- *Пам’ятка про моделювання загроз OWASP* (https://www.owasp.org/index.php/Application_Security_Architecture_Cheat_Sheet);
- *Пам’ятка про аналіз поверхні атак OWASP* (https://www.owasp.org/index.php/Attack_Surface_Analysis_Cheat_Sheet).

OB2. Вимоги щодо оцінювання відповідності автентифікації

Задача

Автентифікація є актом встановлення або підтвердження чогось або когось як автентичного, тобто засвідчення достовірності тверджень, зроблених кимось чи про щось.

Забезпечення відповідності оцінюваного додатка наступним високорівневим вимогам:

- підтвердження цифрової особистості ініціатора комунікації (відправника);
- забезпечення можливості автентифікації виключно авторизованих осіб, а також захищена передача облікових даних.

Вимоги

№	Опис	1	2	3	З версії
2.1	Перевірити, чи всі сторінки та ресурси вимагають автентифікації за замовченням, за винятком тих, які призначені саме для того, щоб бути відкритими (принцип повного посередництва).	✓	✓	✓	1.0
2.2	Перевірити, чи поля паролів не відображають введений користувачем пароль.	✓	✓	✓	1.0
2.4	Перевірити, чи на серверній стороні примусово застосовуються всі контролі автентифікації.	✓	✓	✓	1.0
2.6	Перевірити, чи контролі автентифікації забезпечують захист від входу зломщиків в систему.	✓	✓	✓	1.0
2.7	Перевірити, чи поля для введення паролю дозволяють і заохочують використання парольних фраз і не забороняють використання довгих парольних фраз / дуже складних паролів.	✓	✓	✓	3.0
2.8	Перевірити, чи всі функції автентифікації особистості (як, наприклад, редагування профілю, забутий пароль, вимкнений / втрачений маркер доступу, служба технічної підтримки або *IVR*[20]), які можуть відновити доступ до облікового запису, не менш стійкі до атак, ніж механізм первинної автентифікації.	✓	✓	✓	2.0
2.9	Перевірити, чи функціональна можливість зміни паролю вимагає старий пароль, новий пароль і підтвердження паролю.	✓	✓	✓	1.0
2.12	Перевірити, чи реєструються всі підозрілі дії щодо автентифікації зі зазначенням запитів із відповідними метаданими, необхідними для розслідування порушень безпеки.		✓	✓	2.0
2.13	Перевірити, чи паролі облікових записів зберігаються з використанням достатньо сильного шифрування та відбивають атаки методом перебору на процедуру шифрування.		✓	✓	3.0
2.16	Перевірити, чи ідентифікаційні дані транспортуються по відповідному зашифрованому каналу, а також чи всі сторінки / функції, які вимагають введення користувачем його ідентифікаційних даних, використовують	✓	✓	✓	3.0

[20] IVR (Interactive Voice Response) – інтерактивна голосова відповідь.

	зашифровані канали.				
2.17	Перевірити, чи функція відновлення забутого паролю та інші шляхи його відновлення не відображають діючий пароль, а також чи користувач не отримує новий пароль у вигляді незашифрованого тексту.	✓	✓	✓	2.0
2.18	Перевірити, що не можливий збір даних через функціональні можливості входу в систему, зміни паролю або відновлення забутого облікового запису.	✓	✓	✓	2.0
2.19	Перевірити, чи в програмному середовищі додатка або в будь-якому з його компонентів не застосовуються типові паролі (як, наприклад, *admin*[21] чи *password*[22]).	✓	✓	✓	2.0
2.20	Перевірити наявність регулювання кількості запитів для попередження автоматизованих атакз метою автентифікації як, наприклад, атаки методом перебору або атаки на відмову в обслуговуванні.	✓	✓	✓	3.0
2.21	Перевірити, чи всі автентифікаційні дані для отримання доступу до зовнішніх служб є зашифрованими і зберігаються у захищеному місці.		✓	✓	2.0
2.22	Перевірити, чи функція відновлення забутого паролю та інші шляхи його відновлення застосовують додатковий програмний засіб автентифікації, повідомлення на мобільний телефон або механізм відновлення без використання мережі.	✓	✓	✓	3.0
2.23	Перевірити, чи блокування облікового запису має статуси як м'якого, так і жорсткого блокування, які не є взаємовиключними. Якщо обліковий запис тимчасово заблокований у м'якому режимі внаслідок атаки методом перебору, статус жорсткого блокування вимикатися не повинен.		✓	✓	3.0
2.24	Перевірити, чи питання, які ґрунтуються на знаннях користувача (також відомі як секретні питання), якщо такі вимагаються, є достатньо сильними для захисту додатка.	✓	✓	✓	2.0
2.25	Перевірити, чи можна налаштувати систему таким чином, щоб не допускати повторного використання зазначеної кількості минулих паролів.		✓	✓	2.0
2.26	Перевірити, чи вимагається повторна, ступінчаста, адаптивна або двохфакторна автентифікація або підпис транзакції перед дозволом будь-яких ризикованих операцій, пов'язаних із додатком, згідно з профілем ризиків додатка.		✓	✓	2.0
2.27	Перевірити наявність заходів для блокування використання розповсюджених паролів і слабких парольних фраз.	✓	✓	✓	3.0
2.28	Перевірити, чи додаток відповідає на всі виклики автентифікації, як успішні, так і безуспішні, протягом однакового середнього часу очікування.			✓	3.0
2.29	Перевірити, чи секретні коди, ключі програмного інтерфейсу додатка (API) та паролі не містяться у вихідному коді або мережевих репозитаріях вихідного коду.			✓	3.0
2.30	Перевірити, що за можливості автентифікації користувачів додаток використовує надійний механізм безпеки автентифікації.	✓	✓	✓	3.0
2.31	Перевірити, що за можливості автентифікації користувачів додаток		✓	✓	3.0

[21] *Admin* в перекладі з англійської мови означає «адмін», «адміністратор».
[22] *Password* в перекладі з англійської мови означає «пароль».

	дозволяє застосування двохфакторної або іншої стійкої автентифікації чи будь-якої аналогічної схеми, яка захищає ім'я користувача та пароль від розкриття.				
2.32	Перевірити, що інтерфейси адміністратора не доступні для сторін, які не заслуговують на довіру.	✓	✓	✓	3.0

Посилання

Детальнішу інформацію наведено в наступних документах:

- *Настанови щодо тестування 4.0: тестування автентифікації OWASP* (https://www.owasp.org/index.php/Testing_for_authentication);
- *Пам'ятка про сховище паролів OWASP* (https://www.owasp.org/index.php/Password_Storage_Cheat_Sheet);
- *Пам'ятка про забуті паролі OWASP* (https://www.owasp.org/index.php/Forgot_Password_Cheat_Sheet);
- *Вибір і застосування контрольних запитань OWASP* *https://www.owasp.org/index.php/Choosing_and_Using_Security_Questions_Cheat_Sheet* (див. *Список дій*).

ОВ3. Вимоги щодо оцінювання відповідності управління сеансами

Задача

Одним із основних компонентів будь-якого мережевого додатка є механізм, який контролює його та підтримує у стані, за якого користувач може взаємодіяти з ним. Це називається управлінням сеансами та описується як набір усіх контролів, які керують повною взаємодією між користувачем і мережевим додатком.

Забезпечення відповідності оцінюваного додатка наступним високорівневим вимогам управління сеансами:

- сеанси є унікальними для кожної окремої особи, їх не можна вгадати та ними не можна користуватися спільно;
- сеанси перериваються, коли є непотрібними, та блокуються внаслідок перевищення часу очікування в періоди неактивності.

Вимоги

№	Опис	1	2	3	З версії
3.1	Перевірити наявність диспетчера сеансів користувачів і його стійкість до всіх типових атак на управління сеансами.	✓	✓	✓	1.0
3.2	Перевірити, чи сеанси перериваються при виході користувача зі системи.	✓	✓	✓	1.0
3.3	Перевірити, чи сеанси блокуються внаслідок перевищення часу очікування через певний період неактивності.	✓	✓	✓	1.0
3.4	Перевірити, чи сеанси блокуються внаслідок перевищення максимально допустимої адміністратором тривалості сеансу незалежно від активності користувача (абсолютне блокування внаслідок перевищення максимальної тривалості сеансу).			✓	1.0
3.5	Перевірити, чи всі сторінки, які вимагають автентифікації, мають простий і видимий доступ до функціональної можливості виходу зі системи.	✓	✓	✓	1.0
3.6	Перевірити, чи ідентифікатори сеансів ніколи не відображаються в *URL*[23]-адресах, повідомленнях про помилки або журналах, а також чи додаток не підтримує перевизначення *URL* кукі-файлів сеансів.	✓	✓	✓	1.0
3.7	Перевірити, чи при кожній успішній автентифікації та повторній автентифікації створюється новий сеанс та ідентифікатор сеансу.	✓	✓	✓	1.0
3.10	Перевірити, чи додаток визнає активними лише ті ідентифікатори сеансу, які генеруються його програмним середовищем.		✓	✓	1.0
3.11	Перевірити, чи ідентифікатори сеансів є достатньо довгими, випадковими та унікальними по всій базі коректних активних сеансів.	✓	✓	✓	1.0

[23] *URL* (*Uniform Resource Locator*) – уніфікований локатор ресурсів.

3.12	Перевірити, чи ідентифікатори сеансів, що зберігаються в кукі-файлах, мають шлях, який веде до достатньо обмеженого для додатка значення, а також, що маркери сеансу автентифікації додатково задають атрибути *HttpOnly*[24] та *secure*[25].	✓	✓	✓	3.0
3.16	Перевірити, чи обмежує додаток кількість паралельних активних сеансів.	✓	✓	✓	3.0
3.17	Перевірити, чи список активних сеансів відображається у профілі облікового запису кожного користувача або в чомусь аналогічному. Користувач повинен мати змогу перервати будь-який активний сеанс.	✓	✓	✓	3.0
3.18	Перевірити, чи користувачу рекомендується перервати всі активні сеанси після успішної зміни паролю.	✓	✓	✓	3.0

Посилання

Детальнішу інформацію наведено в наступних документах:

- *Настанови щодо тестування 4.0: тестування управління сеансами OWASP* (https://www.owasp.org/index.php/Testing_for_Session_Management);
- *Пам'ятка про управління сеансами OWASP* (https://www.owasp.org/index.php/Session_Management_Cheat_Sheet).

[24] *HttpOnly* в перекладі з англійської мови означає «лише протокол передачі гіпертекстових документів».
[25] *Secure* в перекладі з англійської мови означає «безпечний».

ОВ4. Вимоги щодо оцінювання відповідності контролю доступу

Задача

Концепція авторизації полягає в наданні доступу до ресурсів лише тим особам, яким дозволено використовувати їх.

Забезпечення відповідності оцінюваного додатка наступним високорівневим вимогам:

- особи, які мають доступ до ресурсів, надають достовірні ідентифікаційні дані;
- користувачі отримують чіткі ролі та привілеї;
- метадані щодо ролей і дозволів захищені від повторення та втручання.

Вимоги

№	Опис	1	2	3	З версії
4.1	Перевірити наявність принципу мінімальних привілеїв: користувачам надається доступ лише до функцій, інформаційних файлів, *URL*-адрес, контролерів, послуг та інших ресурсів, для яких у них є спеціальний дозвіл на доступ. Це означає захист від підміни ідентифікатора користувача та підвищення привілеїв.	✓	✓	✓	1.0
4.4	Перевірити, чи доступ до конфіденційних записів захищений належним чином, коли кожен користувач має доступ тільки до авторизованих об'єктів і даних (наприклад, захист від втручання інших користувачів за допомогою параметра перегляду чи внесення змін в обліковий запис іншого користувача).	✓	✓	✓	1.0
4.5	Перевірити, чи деактивовано функцію перегляду каталогів, якщо її не активовано навмисно. Окрім того, додатки не повинні мати дозволу на виявлення чи розкриття метаданих файлів і каталогів, як, наприклад, папок *Thumbs.db*, *.DS_Store*, *.git* або *.svn*.	✓	✓	✓	1.0
4.8	Перевірити що контролі доступу у разі відмови залишають додаток в захищеному стані (не дозволяють доступ).	✓	✓	✓	1.0
4.9	Перевірити, чи застосовуються однакові правила контролю доступу як на рівні представлення, так і на серверній стороні.	✓	✓	✓	1.0
4.10	Перевірити, чи кінцеві користувачі не можуть змінити всі атрибути і дані користувача, а також інформацію про політику, яку використовують контролі доступу, якщо в них немає на це спеціального дозволу.		✓	✓	1.0
4.11	Перевірити наявність централізованого механізму (включаючи бібліотеки, які вимагають зовнішньої авторизації) для захисту доступу до всіх типів захищених ресурсів.			✓	1.0
4.12	Перевірити можливість реєстрування всіх рішень, пов'язаних із контролем доступу, в тому числі безуспішних.		✓	✓	2.0
4.13	Перевірити, чи додаток і програмне середовище використовують сильні випадкові анти-*CSRF*[26]-маркери або мають інший механізм захисту транзакцій.	✓	✓	✓	2.0

[26] *CSRF* (*Cross-Site Request Forgery*) – підробка міжсайтових запитів.

4.14	Перевірити, чи система може захищати від агрегованого або тривалого доступу до захищених функцій, ресурсів або даних. Наприклад, розглянути застосування регулятора ресурсів з метою обмеження кількості правок за годину або попередження зчитування всієї бази даних окремим користувачем.		✓	✓	2.0
4.15	Перевірити, чи додаток має додаткову авторизацію (наприклад, ступінчасту або адаптивну автентифікацію) для малоцінних систем та / або розподіл обов'язків для додатків особливої важливості для застосування антишахрайських контролів з урахуванням ризиків додатка та минулих шахрайських дій.		✓	✓	3.0
4.16	Перевірити, чи додаток здійснює контекстозалежну авторизацію правильно, забороняючи неавторизовані маніпуляції шляхом втручання в параметри.	✓	✓	✓	3.0

Посилання

Детальнішу інформацію наведено в наступних документах:

- *Настанови щодо тестування 4.0: авторизація OWASP* (https://www.owasp.org/index.php/Testing_for_Authorization);
- *Пам'ятка про контроль доступу OWASP* (https://www.owasp.org/index.php/Access_Control_Cheat_Sheet).

ОВ5. Вимоги щодо оцінювання відповідності обробки шкідливих вхідних даних

Задача

Найрозповсюдженішою слабкою стороною веб-безпеки додатка є його неспроможність належним чином перевіряти перед використанням вхідні дані, що поступають від клієнта або зі середовища. Це спричиняє абсолютну більшість основних вразливостей веб-додатків, як, наприклад, міжсайтове виконання сценаріїв, *SQL*-ін'єкції, ін'єкції команд інтерпретатора, атаки на Локаль та символи *Юнікод*, атаки на файлові системи і переповнення буфера.

Забезпечення відповідності оцінюваного додатка наступним високорівневим вимогам:

- усі вхідні дані перевірені на правильність та придатність для їх цільового використання;
- ніколи не можна довіряти даним, отриманим із зовнішніх джерел або клієнтів; обробляти такі дані необхідно відповідним чином.

Вимоги

№	Опис	1	2	3	З версії
5.1	Перевірити, чи середовище виконання не вразливе до переповнення буфера, а також чи контролі безпеки запобігають переповненню буфера.	✓	✓	✓	1.0
5.3	Визначити, чи реєструються та відхиляються запити після помилки серверної перевірки вхідних даних.	✓	✓	✓	1.0
5.5	Визначити, чи застосовуються процедури перевірки вхідних даних на серверній стороні.	✓	✓	✓	1.0
5.6	Визначити, чи додаток застосовує єдині контролі перевірки вхідних даних для всіх типів прийнятих даних.			✓	1.0
5.10	Перевірити, чи *SQL*-запити, *HQL*[27], *OSQL*[28], *NoSQL*[29], збережені процедури та виклики збережених процедур захищені за допомогою попередньо підготовлених операторів або параметризації запитів і, відповідно, не вразливі до *SQL*-ін'єкцій.	✓	✓	✓	2.0
5.11	Перевірити, чи додаток не вразливий до *LDAP*-ін'єкцій, а також чи контролі безпеки запобігають *LDAP*-ін'єкціям.	✓	✓	✓	2.0
5.12	Перевірити, чи додаток не вразливий до ін'єкції *OS*[30]-команд, а також чи контролі безпеки запобігають ін'єкціям *OS*-команд.	✓	✓	✓	2.0
5.13	Перевірити, чи додаток не вразливий до дистанційного або локального включення файлів при використанні вмісту, який є шляхом до файлу.	✓	✓	✓	3.0

[27] *HQL* (*Hibernate Query Language*) – мова запитів гібернейт.
[28] *OSQL* (*Object-Structured Query Language*) – об'єктно-орієнтований *SQL*.
[29] *NoSQL* (*not only SQL*) в перекладі з англійської мови означає «не тільки *SQL*».
[30] *OS* (*operating system*) – операційна система.

5.14	Перевірити, чи додаток не вразливий до розповсюджених *XML*[31]-атак, як, наприклад, до втручання в *XPath*[32]-запити, *XXE*[33]-атак та *XML*-ін'єкцій.	✓	✓	✓	2.0
5.15	Переконатися, що всі рядкові змінні, які містяться в *HTML* або в іншому коді веб-клієнта, належним чином контекстуально закодовані вручну або застосовують шаблони автоматичного контекстуального кодування з метою забезпечення невразливості додатка до атак через активне *XSS*, пасивне *XSS* та *XSS* в *DOM*[34].	✓	✓	✓	2.0
5.16	Перевірити, чи чутливі поля, як, наприклад, *accountBalance*[35], *role*[36] або *password*, захищені від шкідливого автоматичного прив'язування, якщо програмне середовище додатка дозволяє автоматичне масове присвоєння параметрів, яке ще називають автоматичним прив'язуванням змінних, вхідного запиту моделі.		✓	✓	2.0
5.17	Перевірити, чи додаток захищений від атак забруднення параметра *HTTP*[37], зокрема, якщо програмне середовище додатка не розпізнає джерело параметрів запиту (*GET*[38], *POST*[39], кукі, заголовки, середовище тощо).		✓	✓	2.0
5.18	Визначити, чи, окрім серверної, застосовується клієнтська перевірка як друга лінія захисту.		✓	✓	3.0
5.19	Визначити, чи всі вхідні дані (не лише поля форми *HTML*, а й усі джерела вхідних даних, як, наприклад, *REST*-виклики, параметри запитів, *HTTP*-заголовки, кукі, командні файли, *RSS*[40]-канали тощо) спершу перевіряються за допомогою позитивної перевірки (внесення в білий список), а потім – слабших форм перевірки, як, наприклад, внесення в сірий список (відфільтровуванння завідомо поганих рядків) або відхилення неправильних вхідних даних (внесення в чорний список).		✓	✓	3.0
5.20	Визначити, чи структуровані дані строго типізовані та перевірені згідно з визначеною схемою, включаючи допустимі символи, довжину та шаблони (наприклад, номери кредитних карток / телефонів або перевірка обґрунтованості пов'язання двох полів, як, наприклад, перевірка відповідності околиці її поштовому індексу).		✓	✓	3.0
5.21	Перевірити, чи в рамках застосування типових заходів безпеки «підчищаються» неструктуровані дані, як, наприклад, допустимі символи та довжина, а також чи екрануються символи, які є потенційно небезпечними в певному контексті (наприклад, імена в *Юнікоді* або з апострофом, як, наприклад, *ʻaʻa* або *O'Hara*).		✓	✓	3.0
5.22	Визначити, чи в рамках перевірки та шифрування вхідних даних сумнівні *HTML* з редактора *WYSIWYG*[41] чи будь-яких інших аналогічних редакторів «підчищаються» за допомогою *HTML*-дезінфектора та обробляються належним чином.	✓	✓	✓	3.0

[31] *XML* (*Extensible Markup Language*) – розширювана мова розмітки.
[32] *XPath* (*XML Path Language*) – мова виразів для визначення частини *XML*-документа або для обчислення на основі вмісту *XML*-документа.
[33] *XXE* (*XML External Entity*) – зовнішня сутність *XML*.
[34] *DOM* (*Document Object Model*) – об'єктна модель документа.
[35] *accountBalance* в перекладі з англійської мови означає «баланс рахунку».
[36] *role* в перекладі з англійської мови означає «роль».
[37] *HTTP* (*HyperText Transfer Protocol*) – протокол передачі гіпертекстових документів
[38] *GET* в перекладі з англійської мови означає «отримати». За допомогою запитів типу *GET* запрошується вміст вказаного ресурсу.
[39] *POST* в перекладі з англійської мови означає «відправити поштою». За допомогою запитів типу *POST* заданому ресурсу передаються призначені для користувача дані.
[40] *RSS* (*Really Simple Syndication*) – дуже просте отримання зведеної інформації.
[41] *WYSIWYG* (*What You See Is What You Get*) в перекладі з англійської мови означає «що бачиш, те й отримуєш».

5.23	Перевірити підключення «підчищення» *HTML* у випадку шаблонної технології автоекранування символів, якщо екранування символів інтерфейсу користувача вимкнене.	✓	✓	3.0
5.24	Перевірити, чи при передачі даних з одного *DOM*-контексту в інший застосовуються безпечні *JavaScript*[42]-методи, як, наприклад, *.innerText* і *.val*.	✓	✓	3.0
5.25	Перевірити, чи при здійсненні *JSON*[43]-інтерпретації у браузерах на клієнті застосовується метод *JSON.parse*[44]. Непотрібно застосовувати функцію *eval*[45]*()* для здійснення *JSON*-інтерпретації на клієнті.	✓	✓	3.0
5.26	Визначити, чи дані аутентифікації очищаються з клієнтського сховища після завершення сеансу, як, наприклад, з *DOM*-браузера.	✓	✓	3.0

Посилання

Детальнішу інформацію наведено в наступних документах:

- *Настанови щодо тестування 4.0: тестування перевірки вхідних даних OWASP*
- (https://www.owasp.org/index.php/Testing_for_Input_Validation);
- *Пам'ятка про перевірку вхідних даних OWASP* (https://www.owasp.org/index.php/Input_Validation_Cheat_Sheet);
- *Настанови щодо тестування 4.0: тестування на забруднення параметра HTTP OWASP* (https://www.owasp.org/index.php/Testing_for_HTTP_Parameter_pollution_%28OTG-INPVAL-004%29);
- *Пам'ятка про LDAP-ін'єкцію OWASP* (https://www.owasp.org/index.php/LDAP_Injection_Prevention_Cheat_Sheet);
- *Настанови щодо тестування 4.0: тестування клієнтської сторони OWASP* (https://www.owasp.org/index.php/Client_Side_Testing);
- *Пам'ятка про попередження міжсайтового виконання сценаріїв OWASP* (https://www.owasp.org/index.php/XSS_%28Cross_Site_Scripting%29_Prevention_Cheat_Sheet);
- Проект *OWASP Java*[46]*-кодування* (https://www.owasp.org/index.php/OWASP_Java_Encoder_Project).

Детальнішу інформацію про автоекранування символів наведено в наступних документах:

- *Зменшення загрози від XSS шляхом автоматичного контекстозалежного екранування символів у шаблонних системах* (http://googleonlinesecurity.blogspot.com/2009/03/reducing-xss-by-way-of-automatic.html);
- *Строго контекстуальне екранування символів AngularJS*[47] (https://docs.angularjs.org/api/ng/service/$sce).

[42] *JavaScript* – динамічна об'єктно-орієнтована мова програмування *JavaScript*.
[43] *JSON* (*JavaScript Object Notation*) – об'єктний запис *JavaScript*.
[44] *Parse* в перекладі з англійської мови означає «інтерпретація».
[45] *eval* (*evaluation*) в перекладі з англійської мови означає «оцінювання».
[46] *Java* – об'єктно-орієнтована мова програмування.
[47] *AngularJS* – каркас *JavaScript* з відкритим програмним кодом.

OB6. Вихідне кодування / екранування символів

Цей розділ було включено до Розділу OB5 у *Стандарті оцінювання відповідності безпеки додатків 2.0*. Вимога 5.16 *ASVS* стосується контекстуального вихідного кодування з метою попередження міжсайтового виконання сценаріїв.

ОВ7. Вимоги щодо оцінювання відповідності криптографії при зберіганні

Задача

Забезпечення відповідності оцінюваного додатка наступним високорівневим вимогам:

- забезпечення захисту у випадку відмови всіх криптографічних модулів, правильна обробка помилок;
- застосування відповідного генератора випадкових чисел за необхідності забезпечення довільного вибору;
- безпечність управління доступом до ключів.

Вимоги

№	Опис	1	2	3	З версії
7.2	Перевірити, чи забезпечується захист у випадку відмови всіх криптографічних модулів, а також чи обробка помилок не призводить до атаки доповнення на *Oracle*[48].	✓	✓	✓	1.0
7.6	Перевірити, чи всі випадкові числа, назви файлів, *GUID*[49] і рядки генеруються за допомогою затвердженого генератора випадкових чисел криптографічного модуля, щоб жоден зломщик не міг вгадати ці випадкові значення.		✓	✓	1.0
7.7	Перевірити, чи криптографічні алгоритми, які застосовує додаток, відповідають *FIPS 140-2*[50] або аналогічному стандарту.	✓	✓	✓	1.0
7.8	Перевірити, чи криптографічні модулі використовуються в затвердженому режимі згідно з політикою безпеки.			✓	1.0
7.9	Перевірити наявність чіткої політики управління криптографічними ключами (наприклад, генерування, розповсюдження, анулювання та закінчення чинності). Перевірити правильність життєвого циклу таких ключів.		✓	✓	1.0
7.11	Перевірити заборону прямого доступу користувачів криптографічних послуг до матеріалу для ключів. Ізолювати криптографічні процеси, у тому числі мастер-секрети, та розглянути можливість використання апаратного сховища ключів (*HSM*[51]).			✓	3.0
7.12	Перевірити, чи інформація, що дозволяє ідентифікувати особистість, є зашифрованою при зберіганні, а також чи передача даних здійснюється через захищені канали.		✓	✓	3.0
7.13	Перевірити, чи ключі та секрети по можливості встановлюються в нульовий стан при знищенні.		✓	✓	3.0
7.14	Перевірити, чи допускається заміна всіх ключів і паролів, а також чи вони генеруються або заміняються під час установки.		✓	✓	3.0

[48] *Oracle* – об'єктно-реляційна система керування базами даних.
[49] *GUID* (*Globally Unique Identifier*) – глобально унікальний ідентифікатор – унікальний реєстраційний номер, що використовується як ідентифікатор програмного засобу.
[50] *FIPS 140-2* (*The Federal Information Processing Standard Publication 140-2*) – *Публікація 140-2 Федерального стандарту з обробки інформації*.
[51] *HSM* (*Hierarchical Storage Management*) – ієрархічне управління носіями.

7.15	Перевірити, чи випадкові числа створюються з відповідною ентропією навіть при високому навантаженні на додаток, або чи додаток значно втрачає свою функціональність за таких умов.	✓	3.0

Посилання

Детальнішу інформацію наведено в наступних документах:

- *Настанови щодо тестування 4.0: тестування на слабку криптографію OWASP* (https://www.owasp.org/index.php/Testing_for_weak_Cryptography);
- *Пам'ятка про криптографічні сховища OWASP* (https://www.owasp.org/index.php/Cryptographic_Storage_Cheat_Sheet).

OB8. Вимоги щодо оцінювання відповідності обробки та журналів реєстрації помилок

Задача

Основною задачею при обробці та реєстрації помилок є забезпечення необхідної реакції користувачів, адміністраторів і груп реагування на інциденти. Задача полягає не у створенні величезної кількості зареєстрованих помилок, а в їх високоякісній реєстрації, яка надає більше інформації ніж створює гамір.

Високоякісні журнали реєстрації помилок часто містять конфіденційні дані, тому їх необхідно захищати законами або директивами про конфіденційність персональних даних, у тому числі:

- не збирати та не реєструвати конфіденційну інформацію, якщо це не вимагається;
- безпечно обробляти всі зареєстровані дані та належним чином захищати їх у відповідності до ступеня їх конфіденційності;
- не зберігати журнали реєстрації вічно: абсолютна тривалість їх зберігання повинна бути максимально короткою.

Якщо журнали реєстрації містять персональні чи конфіденційні дані (визначення останніх може різнитися залежно від країни), вони стають чи не найбільш важливою інформацією в додатку і, відповідно, є привабливими для зломщиків.

Вимоги

№	Опис	1	2	3	З версії
8.1	Перевірити, чи додаток не видає повідомлення про помилки або траси стеків, що містять конфіденційні дані, які можуть бути корисними для зломщиків, включаючи ідентифікатори сеансів, версії програмного забезпечення / програмного середовища та персональні дані.	✓	✓	✓	1.0
8.2	Перевірити, чи логічні схеми обробки помилок у рамках контролю безпеки відмовляють у доступі за промовчанням.		✓	✓	1.0
8.3	Перевірити, чи контролі журналів реєстрації помилок системи безпеки надають можливість реєструвати успішні події, а особливо часткові помилки, які визначаються як такі, що є важливими для системи безпеки.		✓	✓	1.0
8.4	Перевірити, чи кожна подія в журналі реєстрації помилок містить усю інформацію, яка може знадобитися при детальному дослідженні хронології події.		✓	✓	1.0
8.5	Перевірити, чи події, які містять неперевірені дані, не виконуватимуться як код у цільовому програмному забезпеченні, що проглядає журнали реєстрації помилок.			✓	1.0
8.6	Перевірити, чи журнали реєстрації помилок системи безпеки захищені від неавторизованого доступу та несанкціонованого внесення змін.		✓	✓	1.0

8.7	Перевірити, чи додаток не реєструє конфіденційні дані, що визначаються як такі згідно зі законами або настановами щодо конфіденційності персональних даних, конфіденційні дані організації, що визначаються як такі після оцінювання ризиків, або конфіденційні автентифікаційні дані, які можуть бути корисними для зломщиків, включаючи ідентифікатори сеансів користувачів, паролі, хеші або *API*[52]-маркери.	✓	✓	3.0
8.8	Перевірити, чи всі недруковані символи та розділювачі полів належним чином закодовані в записах журналів, щоб попередити ін'єкції журналів.		✓	2.0
8.9	Перевірити, чи розрізняються в записах журналів поля з достовірних і неперевірених джерел.		✓	2.0
8.10	Перевірити, чи журнали аудиту або аналогічні журнали забезпечують неспростовність основних транзакцій.	✓	✓	3.0
8.11	Перевірити, чи журнали реєстрації помилок системи безпеки оснащені якимось механізмом перевірки або контролями цілісності для попередження несанкціонованого внесення змін.		✓	3.0
8.12	Перевірити, чи журнали зберігаються не в тому розділі, в якому працює додаток, і чи ротація журналу є правильною.		✓	3.0

Посилання

Детальнішу інформацію наведено в наступному документі:

- *Настанови щодо тестування 4.0: тестування на обробку помилок OWASP* (https://www.owasp.org/index.php/Testing_for_Error_Handling).

[52] *API* (*Application Programming Interface*) – прикладний програмний інтерфейс.

ОВ9. Вимоги щодо оцінювання відповідності захисту даних

Задача

Три основні критерії надійного захисту інформації – конфіденційність, цілісність і доступність (тріада *CIA*[53]). Цей стандарт ґрунтується на тому, що захист інформації здійснюється в безпечній системі, як, наприклад, захищений сервер із відповідними засобами безпеки. Коли йдеться про додаток, необхідно допускати таку можливість, що користувацькі пристрої можуть бути певною мірою скомпрометованими. Якщо додаток передає чи зберігає конфіденційну інформацію на незахищених пристроях, як, наприклад, на чужих комп'ютерах, телефонах чи планшетах, додаток несе відповідальність за шифрування даних, які зберігаються на таких пристроях, а також за їх захист від незаконного отримання, зміни та розкриття.

Забезпечення відповідності оцінюваного додатка наступним високорівневим вимогам захисту інформації:

- конфіденційність (дані повинні бути захищеними від неавторизованого перегляду та розкриття як при передачі, так і при зберіганні);
- цілісність (дані повинні бути захищеними від зловмисного створення, змінення та видалення неавторизованими зломщиками);
- доступність (за необхідності дані повинні бути доступними для авторизованих користувачів).

Вимоги

№	Опис	1	2	3	З версії
9.1	Перевірити, чи заборонено кешування всіх форм, які містять конфіденційну інформацію, з клієнтської сторони, включаючи функцію автозаповнення.	✓	✓	✓	1.0
9.2	Перевірити наявність переліку конфіденційних даних, які обробляються додатком, а також чіткої політики щодо їх контролю, шифрування та застосування згідно з відповідними директивами про захист інформації.			✓	1.0
9.3	Перевірити, чи всі конфіденційні дані відправляються на сервер в тілі або заголовку *HTTP*-повідомлень (тобто, чи для відправки конфіденційних даних ніколи не застосовуються *URL*-параметри).	✓	✓	✓	1.0
9.4	Перевірити, чи додаток встановлює відповідні антикеш-заголовки, враховуючи ризики, які стоять перед додатком, як, наприклад, наступні: *Expires: Tue, 03 Jul 2001 06:00:00 GMT* *Last-Modified: {now} GMT* *Cache-Control: no-store, no-cache, must-revalidate, max-age=0*	✓	✓	✓	1.0

[53] *CIA* (*Confidentiality, Integrity and Availability*) – конфіденційність, цілісність і доступність.

	Cache-Control: post-check=0, pre-check=0 *Pragma: no-cache*[54]				
9.5	Перевірити, чи всі кеш- або тимчасові конфіденційні дані, що зберігаються на сервері, захищені від неавторизованого доступу та видаляються / денонсуються після доступу до них авторизованого користувача.		✓	✓	1.0
9.6	Перевірити, чи існує метод видалення конфіденційних даних будь-якого типу з додатка після завершення політики обов'язкового зберігання.			✓	1.0
9.7	Перевірити, чи додаток мінімізує кількість параметрів у запитах, як, наприклад, приховані поля, *Ajax*[55]-змінні, кукі та значення заголовків.		✓	✓	2.0
9.8	Перевірити, чи додаток може виявляти та сповіщати про аномально високу кількість запитів на збір даних, наприклад, як при типовому аналізу екранних даних.			✓	2.0
9.9	Перевірити, чи дані, що зберігаються в клієнтському сховищі, як, наприклад, у локальному сховищі *HTML5*, сховищі сеансів, звичайних куках, *IndexedDB*[56], або флеш-куках, не містять конфіденційних чи особистих ідентифікаційних даних.	✓	✓	✓	3.0
9.10	Перевірити, чи реєструється доступ до конфіденційних даних, якщо дані збираються згідно з відповідними директивами про захист інформації, або чи є необхідність реєстрації доступу.		✓	✓	3.0
9.11	Перевірити, чи конфіденційні дані швидко «підчищаються» з пам'яті, коли стають більше непотрібними, а також чи вони обробляються відповідно до функцій і технік, що підтримуються програмним середовищем / бібліотекою / операційною системою.		✓	✓	3.0

Посилання

Детальнішу інформацію наведено в наступному документі:

- *Пам'ятка про захист конфіденційності користувачів* OWASP (https://www.owasp.org/index.php/User_Privacy_Protection_Cheat_Sheet).

[54] Припинення функціонування: вт, 03 лип. 2001 р., 06:00:00 *GMT* *Greenwich Mean Time* – середній час за Гринвічем
Востаннє змінено: {зараз} *GMT*
Контроль кеш-пам'яті: зберігання заборонено, кешування заборонено, обов'язкова повторна перевірка, максимальний вік=0
Контроль кеш-пам'яті: постперевірка=0, попередня перевірка=0
Прагма: кешування заборонено

[55] *AJAX* (*Asynchronous JavaScript and XML*) – підхід до побудови користувацьких інтерфейсів веб-додатків, за яких веб-сторінка, не перезавантажуючись, у фоновому режимі надсилає запити на сервер і сама звідти довантажує потрібні користувачу дані.

[56] *IndexedDB* (*Indexed Database*) – індексна база даних.

ОВ10. Вимоги щодо оцінювання відповідності безпеки комунікацій

Задача

Забезпечення відповідності оцінюваного додатка наступним високорівневим вимогам:

- застосування *TLS*[57] при передачі конфіденційних даних;
- постійне застосування криптостійких алгоритмів і шифрів.

Вимоги

№	Опис	1	2	3	З версії
10.1	Перевірити, чи можна побудувати шлях від надійного центру сертифікації до кожного *TLS*-сертифіката сервера, а також перевірити дійсність кожного сертифіката сервера.	✓	✓	✓	1.0
10.3	Перевірити, чи застосовується *TLS* для всіх з'єднань (як зовнішніх, так і внутрішніх), які є автентифікованими або містять конфіденційні дані або функції, та не звертаються до сумнівних або нешифрованих протоколів. Переконатися, що обраний алгоритм є найбільш криптостійкою альтернативою.	✓	✓	✓	3.0
10.4	Перевірити, чи реєструються помилки внутрішніх *TLS*-з'єднань.			✓	1.0
10.5	Переконатися, що шляхи до всіх сертифікатів клієнта будуються та перевіряються за допомогою попередньо налаштованих точок довіри з урахуванням інформації про анулювання.			✓	1.0
10.6	Перевірити, чи всі з'єднання зі зовнішніми системами, що стосуються конфіденційної інформації або функцій, є авторизованими.		✓	✓	1.0
10.8	Перевірити наявність єдиного стандарту впровадження *TLS*, що застосовується додатком, налаштованого на функціонування в санкціонованому режимі використання.			✓	1.0
10.10	Перевірити, чи приколоті відкриті ключі *TLS*-сертифікатівдля промисловихта резервних. Детальнішу інформацію наведено в наступних документах (див. посилання нижче).			✓	3.0
10.11	Перевірити наявність *HSTS*[58]-заголовків у всіх запитах і піддоменах, як, наприклад: *Strict-Transport-Security: max-age=15724800; includeSubdomains*[59]	✓	✓	✓	3.0
10.12	Перевірити, чи додано *URL* промислового веб-сайту до попередньо завантаженого переліку доменів зі строгою безпекою передачі інформації, які підтримуються постачальниками веб-браузерів. Див. посилання нижче.			✓	3.0
10.13	Перевірити, чи застосовуються шифри прямої секретності для запобіганням пасивним зломщиків, які реєструють трафік.	✓	✓	✓	3.0

[57] *TLS* (*Transport Layer Security*) – безпека транспортного рівня.
[58] *HSTS* (*HTTP Strict Transport Security*) – строга безпека передачі інформації через протокол *HTTP*.
[59] Строга безпека передачі інформації: максимальний вік =15724800; включити піддомени

10.14	Перевірити, чи налаштоване та належним чином застосовується відкликання сертифікатів, як, наприклад, *OSCP stapling*[60].	✓	✓	✓	3.0
10.15	Перевірити, чи в усій ієрархії сертифікатів використовуються виключно криптостійкі алгоритми, шифри та протоколи, включаючи кореневі та проміжні сертифікати обраного центру сертифікації.	✓	✓	✓	3.0
10.16	Перевірити, чи налаштування *TLS* відповідають сучасним провідним практикам, зокрема, коли загальні налаштування, шифри та алгоритми є незахищеними.	✓	✓	✓	3.0

Посилання

Детальнішу інформацію наведено в наступних документах:

- *Пам'ятка про TLS* OWASP (https://www.owasp.org/index.php/Transport_Layer_Protection_Cheat_Sheet);
- *Замітки щодо «Затверджених режимів TLS»* (колись *ASVS* посилався на американський стандарт *FIPS 140-2*, але застосування американських стандартів може бути складним, суперечливим або спричиняти плутанину, оскільки *ASVS* є міжнародним стандартом; для забезпечення кращого рівня захисту краще досягти відповідності вимозі 10.8 за допомогою аналізу таких настанов, як, наприклад, https://wiki.mozilla.org/Security/Server_Side_TLS, генерування завідомо правильних конфігурацій, як, наприклад, https://mozilla.github.io/server-side-tls/ssl-config-generator/, а також використання загальновідомих інструментів оцінювання *TLS*, як, наприклад, *sslyze*[61], різноманітних сканерів вразливостей або надійних послуг з онлайн-оцінювання *TLS*; загалом, спостерігається невідповідність цьому розділу, зокрема внаслідок використання застарілих або незахищених шифрів та алгоритмів, нестачі досконалої прямої секретності, застарілих або незахищених *SSL*[62]-протоколів, слабких переважаючих шифрів тощо);
- *Прикріплення сертифікатів* (детальнішу інформацію наведено за наступним посиланням: https://tools.ietf.org/html/rfc7469); суть прикріплення сертифікатів на ключі для їх вироблення та відновлення полягає в безперервності бізнесу (див.: https://noncombatant.org/2015/05/01/about-http-public-key-pinning/);
- *Строга безпека передачі інформації згідно з протоколом попереднього завантаження HTTP* (https://www.chromium.org/hsts).

[60] *OSCP stapling* (*Online Certificate Status Protocol stapling*) – «степлерування» *Онлайн-протоколу статусу сертифіката*.
[61] *sslyze* – інструмент для аналізу *SSL*-конфігурації сервера.
[62] *SSL* (*Secure Sockets Layer*) – рівень захищених сокетів.

ОВ11. Вимоги щодо оцінювання відповідності конфігурацій безпеки *HTTP*

Задача

Забезпечення відповідності оцінюваного додатка наступним високорівневим вимогам:

- належне посилення конфігурацій сервера додатка за промовчанням;
- *HTTP*-відповіді повинні містити безпечний набір символів у типі вмісту заголовка.

Вимоги

№	Опис	1	2	3	З версії
11.1	Перевірити, чи додаток приймає лише визначений набір необхідних методів *HTTP*-запитів, як, наприклад, прийняття *GET* і *POST* і чітке блокування методів, які не застосовуються (наприклад, *TRACE*[63], *PUT*[64] і *DELETE*[65]).	✓	✓	✓	1.0
11.2	Перевірити, чи кожна *HTTP*-відповідь має тип вмісту заголовка, що визначає безпечний набір символів (наприклад, *UTF-8*[66], *ISO 8859-1*[67]).	✓	✓	✓	1.0
11.3	Перевірити, чи автентифікує додаток *HTTP*-заголовки, такі як маркер на пред'явника, додані довіреним проксі-сервером або *SSO*-пристроями.		✓	✓	2.0
11.4	Перевірити, чи застосовується *Політика захисту вмісту CSP V2*[68] для сайтів, вміст яких не повинен переглядатися в прихованому фреймі стороннього виробника.		✓	✓	2.0
11.5	Перевірити, чи *HTTP*-заголовки або будь-яка інша частина *HTTP*-відповіді не розкриває детальну інформацію про версію компонентів системи.	✓	✓	✓	2.0
11.6	Перевірити, чи всі *API*-відповіді містять *X-Content-Type-Options: nosniff* і *Content-Disposition: attachment; filename="api.json"* (або інше відповідне ім'я файлу для типу вмісту).	✓	✓	✓	3.0
11.7	Перевірити, чи застосовується політика *CSP V2* для відключення вбудованої *JavaScript* або здійснення перевірки цілісності вбудованої *JavaScript* за допомогою одноразових псевдовипадкових кодів або хешів *CSP*[69].	✓	✓	✓	3.0
11.8	Перевірити наявність заголовку *X-XSS-Protection: 1; mode=block*.	✓	✓	✓	3.0

[63] *TRACE* в перекладі з англійської мови означає «здійснити трасування». За допомогою запитів типу *TRACE* отриманий запит повертається так, щоб клієнт міг побачити, що саме проміжні сервери додають або змінюють в запиті.
[64] *PUT* в перекладі з англійської мови означає «помістити». За допомогою запитів типу *PUT* вказаний ресурс завантажується на сервер.
[65] *DELETE* в перекладі з англійської мови означає «видалити». За допомогою запитів типу *DELETE* вказаний ресурс видаляється.
[66] *UTF-8* (*Unicode Transformation Format 8*) – формат перетворення *Юнікоду 8*; кодування, що реалізовує представлення *Юнікоду*, сумісне з 8-бітовим кодуванням тексту.
[67] *ISO 8859-1* (*ISO – International Organization for Standardization*) – стандарт Міжнародної організації зі стандартизації *ISO 8859-1: набір символів, призначений для запису західноєвропейських мов*.
[68] *CSP V2* (*Content Security Policy Version 2*) – *Політика захисту вмісту* (версія 2).
[69] *CSP* (*Cryptography Service Provider*) – криптопровайдер.

Посилання

Детальнішу інформацію наведено в наступному документі:

- *Настанови щодо тестування 4.0: тестування на дієслівне спотворення HTTP*

  (https://www.owasp.org/index.php/Testing_for_HTTP_Verb_Tampering_%28OTG-INPVAL-003%29).

ОВ12. Вимоги щодо оцінювання відповідності конфігурацій безпеки

Цей розділ було включено до Розділу ОВ11 у *Стандарті оцінювання відповідності безпеки додатків 2.0*.

ОВ13. Вимоги щодо оцінювання відповідності контролю за шкідливими програмними засобами

Задача

Забезпечення відповідності оцінюваного додатка наступним високорівневим вимогам:

- виявлена зловмисна діяльність обробляється безпечно та належним чином, не впливаючи на решту додатка.

Вимоги

№	Опис	1	2	3	З версії
13.1	Перевірити, чи вся зловмисна діяльність відповідним чином захована в пісочницю, контейнеризується або ізолюється для того, щоб затримати або відвернути атаки на інші додатки.			✓	2.0
13.2	Перевірити, чи при аналізі коду здійснюється пошук шкідливих кодів, закладок, великодніх яєць і логічних помилок.			✓	3.0

ОВ14. Вимоги щодо оцінювання відповідності внутрішньої безпеки

Цей розділ було включено до Розділу ОВ13 у *Стандарті оцінювання відповідності безпеки додатків 2.0*.

ОВ15. Вимоги щодо оцінювання відповідності бізнес-логіки

Задача

Забезпечення відповідності оцінюваного додатка наступним високорівневим вимогам:

- потік бізнес-логіки є послідовним і впорядкованим.

Вимоги

№	Опис	1	2	3	З версії
15.1	Перевірити, чи додаток обробляє потоки бізнес-логіки лише в порядку послідовності, коли всі кроки обробляються в реальному часі, а не безладно, пропускаючи деякі кроки, обробляючи кроки інших користувачів або здійснюючи транзакції занадто швидко.		✓	✓	2.0
15.2	Перевірити, чи додаток має бізнес-обмеження та коректно застосовує їх окремо для кожного користувача з відповідною системою оповіщення, що конфігурується, та автоматизованими реакціями на автоматизовані або незвичайні атаки.		✓	✓	2.0

Посилання

Детальнішу інформацію наведено в наступних документах:

- *Настанови щодо тестування 4.0: тестування бізнес-логіки* OWASP (https://www.owasp.org/index.php/Testing_for_business_logic);
- *Пам'ятка OWASP* (https://www.owasp.org/index.php/Business_Logic_Security_Cheat_Sheet).

ОВ16. Вимоги щодо оцінювання відповідності файлів і ресурсів

Задача

Забезпечення відповідності оцінюваного додатка наступним високорівневим вимогам:

- дані файлу, що не заслуговують на довіру, необхідно обробляти відповідним чином і безпечно;
- дані, отримані з неперевірених джерел, необхідно зберігати не в кореневому каталозі веб-вузла та з обмеженим доступом.

Вимоги

№	Опис	1	2	3	З версії
16.1	Перевірити, чи *URL* здійснює переадресацію та пересилання виключно в дозволене місце призначення з білого списку або показує попередження при переадресації на потенційно небезпечний вміст.	✓	✓	✓	2.0
16.2	Перевірити, чи дані файлу, що не заслуговують на довіру, після поступлення в додаток не застосовуються безпосередньо командами вводу-виводу, зокрема з метою захисту від таких вразливостей, як обхід каталогу, включення локальних файлів, *MIME*[70]-тип файлу та ін'єкцій в *OS*-команди.	✓	✓	✓	2.0
16.3	Визначити, чи файли, отримані з неперевірених джерел, перевіряються на предмет того, чи вони є очікуваного типу, та скануються антивірусними програмами з метою попередження вивантаження завідомо шкідливого вмісту.	✓	✓	✓	2.0
16.4	Перевірити, чи дані, що не заслуговують на довіру, не застосовуються у функціональних можливостях включення, завантаження класів або відбиття з метою захисту від вразливостей, пов'язаних із дистанційним або локальним включенням файлів.	✓	✓	✓	2.0
16.5	Перевірити, чи дані, що не заслуговують на довіру, не застосовуються в міждоменному розповсюджені ресурсів з метою захисту від дистанційного управління довільним вмістом.	✓	✓	✓	2.0
16.6	Визначити, чи файли, отримані з неперевірених джерел, зберігаються не в кореневому каталозі веб-вузла з обмеженим доступом і бажано зі строгою перевіркою.		✓	✓	3.0
16.7	Перевірити, чи веб-сервер або сервер додатка має конфігурації за промовчанням з метою обмеження доступу до дистанційних ресурсів або систем поза межами веб-сервера або сервера додатка.		✓	✓	2.0
16.8	Перевірити, чи код додатка не виконує вивантажені дані, отримані з неперевірених джерел.	✓	✓	✓	3.0

[70] *MIME* (*Multipurpose Internet Mail Extensions*) – багатоцільові розширення для Інтернет-пошти.

16.9	Не застосовувати *Flash*[71], *Active-X*[72], *Silverlight*[73], *NaCl*[74], клієнтську *Java* або інші клієнтські технології, які не вбудовані згідно зі стандартами *W3C*[75] для браузерів.	✓	✓	✓	2.0

Посилання

Детальнішу інформацію наведено в наступних документах:

- *Обробка розширення файлу для конфіденційної інформації* (https://www.owasp.org/index.php/Unrestricted_File_Upload);
- *Необмежене вивантаження файлів* (https://www.owasp.org/index.php/Unrestricted_File_Upload).

[71] *Flash* – мультимедійна та програмна платформа для авторської розробки векторної графіки, анімації, ігор і насичених Інтернет-додатків.
[72] *Active-X* – основа для визначення повторно використовуваних компонентів програмного забезпечення незалежно від мови програмування.
[73] *Silverlight* – розширення до веб-браузерів, яке дозволяє відображати на сторінці анімацію, векторну графіку, а також програвати звук і відео.
[74] *NaCl* (*Networking and Cryptography Library*) – мережева криптографічна бібліотека *Сіль*.
[75] *W3C* (*World Wide Web Consortium*) – *Консорціум Всесвітньої павутини*.

ОВ17. Вимоги щодо оцінювання відповідності мобільних додатків

Задача

Забезпечення відповідності оцінюваного додатка наступним високорівневим вимогам:

- усі серверні контролі, як, наприклад, *API* або веб-служби, повинні мати такий самий рівень контролів безпеки, як сам пристрій;
- активи конфіденційної інформації повинні зберігатися на пристрої безпечно;
- усі конфіденційні дані повинні передаватися з пристрою з урахуванням безпеки транспортного рівня.

Примітка. У версії 3.0 від багатьох вимог відмовилися та видалили їх у зв'язку з тим, що вимоги щодо оцінювання відповідності мобільних додатків дублюють інші вимоги *Стандарту*. Комплексне оцінювання мобільних додатків на кожному рівні вимагає розгляду всіх інших вимог, що застосовуються в *Стандарті*.

Вимоги

№	Опис	1	2	3	З версії
17.1	Перевірити, чи значення ідентифікаторів, які зберігаються на пристрої та можуть витягатися іншими додатками, як, наприклад, *UDID*[76] або номер *IMEI*[77], не застосовуються як маркери автентифікації.	✓	✓	✓	2.0
17.2	Перевірити, чи мобільні додатки не зберігають конфіденційні дані в потенційно нешифрованих загальних ресурсах на пристрої (наприклад, на *SD*-картах[78] або у спільних папках).	✓	✓	✓	2.0
17.3	Перевірити, чи конфіденційні дані, як, наприклад, ключові послідовності, не зберігаються на пристрої незахищеними, навіть у захищених зонах системи.	✓	✓	✓	2.0
17.4	Перевірити, чи секретні ключі, *API*-маркери та паролі генеруються в мобільних додатках динамічно.		✓	✓	2.0
17.5	Перевірити, чи мобільні додатки запобігають витоку конфіденційної інформації (наприклад, при збереженні скриншотів додатка він переходить у фоновий режим, або запис конфіденційної інформації в консолі).		✓	✓	2.0
17.6	Перевірити, чи додаток вимагає мінімальні дозволи на застосування необхідних функціональних можливостей і ресурсів.		✓	✓	2.0
17.7	Перевірити, чи конфіденційний код додатка розміщений у пам'яті непередбачувано (наприклад, *ASLR*[79]).	✓	✓	✓	2.0
17.8	Перевірити наявність техніки антиналагодження, яка застосовується для затримки або запобігання дій зломщиків щодо ін'єкцій			✓	2.0

[76] *UDID* (*Unique Device ID*) – унікальний ідентифікатор пристроїв.
[77] *IMEI* (*International Mobile Equipment Identity*) – міжнародний ідентифікатор мобільного обладнання.
[78] *SD*-карта (*secure digital card*) – цифрова карта пам'яті.
[79] *ASLR* (*Address Space Layout Randomization*) – забезпечення випадковості шарів адресного простору.

	налагоджувачів у мобільні додатки (наприклад, *GDB*[80]).				
17.9	Перевірити, чи додаток не експортує конфіденційну активність, наміри, постачальників вмісту тощо для застосування іншими мобільними додатками того ж пристрою.	✓	✓	✓	2.0
17.10	Перевірити, чи в конфіденційних рядках застосовуються змінні структури як, наприклад, чи перезаписуються номери облікових записів, коли вони більше не використовуються (пом'якшення наслідків атак аналізу пам'яті).			✓	2.0
17.11	Визначити, чи активність додатка, наміри, постачальники вмісту тощо перевіряють усі вхідні дані.	✓	✓	✓	2.0

Посилання

Детальнішу інформацію наведено в наступних документах:

- Проект *OWASP Безпека мобільних додатків* (https://www.owasp.org/index.php/OWASP_Mobile_Security_Project);
- *Пам'ятка iOS[81]-розробникам*: (https://www.owasp.org/index.php/IOS_Developer_Cheat_Sheet).

[80] *GDB* (*GNU Debugger*) – переносимий налагоджувач проекту *GNU*.
[81] *iOS* (*iPhone Operating System*) – мобільна операційна система від американської технологічної компанії *Apple*.

ОВ18. Вимоги щодо оцінювання відповідності веб-служб

Задача

Забезпечення відповідності оцінюваного додатка, який використовує веб-служби згідно з *REST* або *SOAP*[82], наступним високорівневим вимогам:

- належна автентифікація, управління сеансами та авторизація всіх веб-послуг;
- перевірка усіх параметрів вхідних даних, що передаютьсяз нижчого на вищий рівень довіри;
- базова здатність до взаємодії шару веб-служб на основі *SOAP*, що сприяє застосуванню *API*.

Вимоги

№	Опис	1	2	3	З версії
18.1	Перевірити, чи на клієнті та сервері застосовується однаковий стиль шифрування.	✓	✓	✓	3.0
18.2	Перевірити, чи доступ до функцій адміністрування та управління додатком веб-служб є виключно в адміністраторів веб-служб.	✓	✓	✓	3.0
18.3	Перевірити наявність та оцінити відповідність *XML* або *JSON*-схеми до прийняття вхідних даних.	✓	✓	✓	3.0
18.4	Перевірити, чи всі вхідні дані відповідають встановленим обмеженням розміру.	✓	✓	✓	3.0
18.5	Перевірити, чи веб-служби на основі *SOAP* відповідають щонайменше базовому профілю *WS-I*[83].	✓	✓	✓	3.0
18.6	Перевірити наявність сеансової автентифікації та авторизації. Більш детальну інформацію наведено в Розділі ОВ2, ОВ3 та ОВ4. Необхідно уникати застосування статичних *API*-ключів і всього аналогічного.	✓	✓	✓	3.0
18.7	Перевірити, чи *REST*-служби захищені від підробок міжсайтових запитів.	✓	✓	✓	3.0
18.8	Визначити, чи *REST*-служби належним чином перевіряють відповідність типу вхідного вмісту очікуваному, як, наприклад, *application/xml* або *application/json*.		✓	✓	3.0
18.9	Перевірити, чи корисне навантаження повідомлення завірено електронним цифровим підписом з метою забезпечення надійної передачі даних між клієнтом і сервером.		✓	✓	3.0
18.10	Перевірити, чи не існує альтернативних і менш захищених шляхів.		✓	✓	3.0

Посилання

Детальнішу інформацію наведено в наступних документах:

[82] *SOAP* (*Simple Object Access Protocol*) – простий протокол доступу до об'єктів.
[83] *WS-I* (*Web Services-Interoperability*) – взаємодія веб-служб.

- *Пам’ятка про безпеку веб-служб OWASP* (https://www.owasp.org/index.php/Web_Service_Security_Cheat_Sheet);
- *Пам’ятка про тестування безпеки веб-служб OWASP* (https://www.owasp.org/index.php/Web_Service_Security_Testing_Cheat_Sheet).

ОВ19. Конфігурації

Задача

Забезпечити наявність в оцінюваному додатку наступного:

- сучасних бібліотек і платформ;
- безпечних за замовчанням конфігурацій;
- належного посилення, при якому внесення користувачем змін у конфігурації за промовчанням не обов'язково створює чи розкриває слабкі сторони системи безпеки або базових систем.

Вимоги

№	Опис	1	2	3	З версії
19.1	Усі компоненти повинні бути сучасними, з відповідними конфігураціями та версіями системи безпеки, включаючи видалення непотрібних конфігурацій і папок, як, наприклад, зразки додатків, документація по платформі та шаблонні користувачі або користувачі за промовчанням.	✓	✓	✓	3.0
19.2	Комунікації між компонентами, як, наприклад, між сервером додатка та сервером баз даних, повинні бути зашифрованими, зокрема, якщо компоненти знаходяться в різних контейнерах або системах.		✓	✓	3.0
19.3	Комунікації між компонентами, як, наприклад, між сервером додатка та сервером баз даних, повинні бути автентифікованими та здійснюватися обліковим записом із мінімально необхідними привілеями.		✓	✓	3.0
19.4	Перевірити, чи розгортання додатка переміщується в пісочницю, контейнеризується або ізолюється належним чином для того, щоб відстрочити або утримати зломщиків від атак на інші додатки.		✓	✓	3.0
19.5	Перевірити, чи процеси компоновки та розгортання додатка здійснюються безпечно.		✓	✓	3.0
19.6	Визначити, чи авторизовані адміністратори мають можливість перевіряти всі конфігурації системи безпеки на предмет порушення їх цілісності.			✓	3.0
19.7	Перевірити, чи всі компоненти додатка завірені електронним цифровим підписом.			✓	3.0
19.8	Перевірити, чи компоненти третіх сторін розміщені в безпечних сховищах.			✓	3.0
19.9	Перевірити, чи у процесі компоновки всі прапорці безпеки мов системного рівня активовано, як, наприклад, *ASLR*, *DEP*[84] і перевірка безпеки.			✓	3.0

[84] *DEP* (*Data Execution Prevention*) – запобігання виконання даних.

Посилання

Детальнішу інформацію наведено в наступному документі:

- *Настанови щодо тестування 4.0: тестування конфігурацій та управління розгортанням OWASP*

  (https://www.owasp.org/index.php/Testing_for_configuration_management).

Додаток A. Що трапилося з…

Найчастіше питання, яке задають читачі *Стандарту*, – «Що трапилося з…». Ця таблиця показує, що трапилося з нерекомендованими вимогами. У випадку зміни існуючої вимоги або додавання нової вимоги їх можна знайти в деталізованих розділах, а не в цій таблиці.

Початковий №	Опис	Статус	Вилучено	Причина
2.3	Перевірити, чи не перевищена максимальна кількість спроб автентифікації, після якої обліковий запис блокується на певний проміжок часу, достатній для утримання зломщиків від атак методом перебору.	Не рекомендовано.	2.0	Замінено більш комплексною вимогою (2.20).
2.5	Перевірити, чи всі контролі автентифікації (включаючи бібліотеки, які викликають зовнішні служби автентифікації) впроваджуються централізовано.	Об'єднано.	3.0	Переміщено в 1.10 як узагальнення всіх контролів безпеки.
2.10	Перевірити, чи вимагається повторна автентифікація до дозволу операцій додатка, пов'язаних із конфіденційною інформацією.	Не рекомендовано.	2.0	Повторна автентифікація спостерігається настільки рідко, що було вирішено видалити цей контроль.
2.11	Перевірити, чи після визначеного адміністратором проміжку часу закінчується термін дії облікового запису.	Не рекомендовано.	2.0	Абсолютне перевищення ліміту часу та закінчення терміну дії облікового запису переміщено як неефективний контроль.
2.14	Перевірити, чи всі дані облікового запису для доступу до служб, які є зовнішніми щодо додатка, шифруються та зберігаються в безпечному місці (не у вихідному коді).	Оновлено.	2.0	Стала вимогою 2.21.
2.15	Перевірити, чи жодний шкідливий код не впливає на контролі автентифікації, що впроваджують або застосовують код.	Переміщено.	2.0	Переміщено в Розділ ОВ13 «Шкідливий код».
3.8	Перевірити, чи після повторної автентифікації змінюється ідентифікатор сеансу.	Оновлено.	3.0	Включено в 3.7.
3.9	Перевірити, чи при виході зі системи змінюється або видаляється ідентифікатор сеансу.	Оновлено.	3.0	Включено в 3.7.
3.13	Перевірити, чи жодний шкідливий код не впливає на контролі управління сеансами, що впроваджують або застосовують код.	Переміщено.	2.0	Переміщено в Розділ ОВ13 «Шкідливий код».
3.14	Перевірити, чи маркери автентифікованого сеансу, що використовують кукі, захищені застосуванням *HttpOnly*.	Оновлено.	3.0	Переміщено в 3.13.
3.15	Перевірити, чи маркери автентифікованого сеансу, що використовують кукі, захищені	Оновлено.	3.0	Переміщено в 3.13.

	атрибутом безпеки.			
4.2	Перевірити, чи користувачі мають доступ виключно до безпечних *URL*, на який у них є окрема авторизація.	Оновлено.	3.0	Включено в 4.1.
4.3	Перевірити, чи користувачі мають доступ виключно до безпечних файлів даних, на який у них є окрема авторизація.	Оновлено.	3.0	Включено в 4.1.
4.13	Перевірити можливість обходу встановлених на додаток обмежень на введення даних і доступ (як, наприклад, обмеження кількості щоденних операцій або послідовність завдань).	Переміщено.	3.0	Переміщено в Розділ ОВ15 «Бізнес-логіка».
4.15	Перевірити, чи жодний шкідливий код не впливає на контролі доступу, що впроваджують або застосовують код.	Переміщено.	2.0	Переміщено в Розділ ОВ13 «Контроль за шкідливими програмними засобами»
5.2	Перевірити наявність і застосування шаблону позитивної перевірки вхідних даних.	Не рекомендовано.	2.0	Переміщено як занадто складну для впровадження вимогу, зокрема у випадку введення текстових вхідних даних вільного формату.
5.4	Перевірити, чи для всіх джерел введення вхідних даних визначено набір символів, як, наприклад, *UTF-8*.	Не рекомендовано.	3.0	Переміщено як вимогу, занадто складну для впровадження більшістю мов.
5.7	Перевірити, чи реєструються всі помилки перевірки вхідних даних.	Не рекомендовано.	3.0	Переміщено як вимогу, яка би створювала занадто багато марних реєстраційних записів, які ігноруватимуться.
5.8	Визначити, чи до їх перевірки всі вхідні дані канонікалізуються для всіх цільових дешифраторів та інтерпретаторів.	Не рекомендовано.	3.0	Переміщено як вимогу, характерну для *JSP*[85]-технологій типу 1, яка не стосується більшості сучасних каркасів.
5.9	Перевірити, чи жодний шкідливий код не впливає на контролі перевірки вхідних даних.	Переміщено.	2.0	Переміщено в Розділ ОВ13 «Контроль за шкідливими програмними засобами».
5.14	Перевірити, чи середовище виконання не вразливе до *XML*-ін'єкцій, і чи контролі безпеки захищають від них.	Об'єднано.	3.0	Об'єднано з 5.13.
5.15	-- ПУСТА ВИМОГА --	Видалено.	3.0	Ця вимога ніколи не існувала.
5.19	Перевірити наявність окремого контролю безпеки для кожного типу вихідного кодування / екранування символів додатком у цільовому місці призначення.	Об'єднано.	3.0	Переміщено в 1.10 як узагальнення всіх контролів безпеки.
7.1	Перевірити, чи застосовуються всі	Не	3.0	Багато сучасних гнучких і

[85] JSP (Java Server Pages) – серверні сторінки Java.

	криптографічні функції, призначені для захисту секретної інформації від користувачів додатка, на серверній стороні.	рекомендовано.		мобільних додатків відповідають цій вимозі за промовчанням.
7.3	Перевірити, чи всі мастер-секрети захищені від неавторизованого доступу. Мастер-секрет – це дані про додаток, що зберігаються як відкритий текст на диску та застосовуються для захисту доступу до інформації про конфігурації безпеки.	Переміщено.	3.0	Переміщено в 2.29.
7.4	Перевірити, чи при створенні в хеші паролів добавляється «сіль».	Переміщено.	2.0	Переміщено в 2.13.
7.5	Перевірити, чи реєструються помилки криптографічного модуля.	Не рекомендовано.	2.0	Створення непотрібних реєстраційних записів, які ніколи не розглядатимуться, є малоефективним.
7.10	Перевірити, чи жодний шкідливий код не впливає на криптографічні модулі, що впроваджують або застосовують код.	Переміщено.	2.0	Переміщено в Розділ ОВ13.
8.2	Перевірити, чи помилки обробляються на безпечному приладі.		3.0	Не рекомендовано.
8.3	Перевірити, чи всі контролі, що здійснюють реєстрацію, впроваджуються на сервері.	Переміщено.	3.0	Переміщено в 1.13 як більш узагальнюючий архітектурний контроль.
8.9	Перевірити, чи в програмних засобах існує єдина реалізація здійснення реєстрації на рівні додатків.	Переміщено.	3.0	Переміщено в 1.13 як більш узагальнюючий архітектурний контроль.
8.11	Перевірити наявність інструменту аналізу журналів реєстрації помилок, що дозволяє аналітику здійснювати пошук зареєстрованих подій в усіх полях, ґрунтуючись на поєднанні критеріїв пошуку, у форматі записів журналів, який підтримується системою.	Не рекомендовано.	3.0	Переміщено як такий, що не вимагається для безпечного програмного забезпечення.
8.12	Перевірити, чи жодний шкідливий код ОВ ОВ ОВ не впливає на обробку помилок і контролі реєстрації, що впроваджують або застосовують код.	Переміщено.	2.0	Переміщено в Розділ ОВ13 «Контроль за шкідливими програмними засобами».
8.15	Перевірити, чи реєстрація здійснюється до виконання транзакцій. Якщо реєстрація пройшла неуспішно (наприклад, недостатньо місця на диску або немає потрібного дозволу) додаток зберігає свою роботоздатність. Саме тоді виникає необхідність у цілісності та неспростовності.	Не рекомендовано.	3.0	Переміщено як занадто детальний контроль, який можна застосовувати лише в невеликій кількості додатків.
10.2	Перевірити, чи невдалі *TLS*-з'єднання не звертаються до незахищених *HTTP*-з'єднань.	Об'єднано.	3.0	Об'єднано з 10.3.
10.7	Перевірити, чи всі з'єднання зі зовнішніми системами, які містять конфіденційні дані або функції, застосовують обліковий запис, налаштований як такий, що має мінімальні			

	привілеї, необхідні для належного функціонування додатка.			
10.9	Перевірити, чи для всіх з'єднань визначено окреме кодування символів (наприклад, *UTF-8*).			
11.1	Не рекомендовано.			
11.4	Не рекомендовано.			
11.5	Не рекомендовано.			
11.6	Не рекомендовано.			
11.7	Не рекомендовано.			
11.8	Не рекомендовано.			
11.4	Не рекомендовано.			
13.1	Не рекомендовано.			
13.2	Не рекомендовано.			
13.3	Не рекомендовано.			
13.4	Не рекомендовано.			
13.5	Не рекомендовано.			
13.6	Не рекомендовано.			
13.7	Не рекомендовано.			
13.8	Не рекомендовано.			
13.9	Не рекомендовано.			
15.1-15.7, 15.9	Розділ про бізнес-логіку.	Об'єднано.	3.0	Більшу частину Розділу OB15 було об'єднано з 15.8 і 15.10.
15.11	Перевірити, чи додаток охоплює всі ризики, пов'язані зі *STRIDE*.	Продубльовано.	3.0	Продубльована вимога, яку можна знайти в 1.6.
16.4	Визначати, чи параметри, отримані з неперевірених джерел, не застосовуються при обробці імен файлів, повних назв каталогів або всіх інших об'єктів файлової системи без попередньої канонікалізації та перевірки вхідних даних для попередження атак методом локального включення файлів.	Переміщено.	3.0	Переміщено в 16.2.
17.1	Визначити, чи клієнт перевіряє *SSL*-сертифікати.	Не рекомендовано.	3.0	Продубльована вимога. Загальну вимогу можна знайти в Розділі OB10.
17.7	Не рекомендовано.			

17.8	Не рекомендовано.			
17.10	Не рекомендовано.			
17.11	Не рекомендовано.			
17.12	Не рекомендовано.			
17.13	Не рекомендовано.			
17.14	Не рекомендовано.			
17.15	Не рекомендовано.			
17.16	Не рекомендовано.			
17.17	Не рекомендовано.			
17.18	Не рекомендовано.			
17.19	Не рекомендовано.			
17.20	Не рекомендовано.			
17.22	Не рекомендовано.			
17.23	Не рекомендовано.			
17.24	Не рекомендовано.			

Додаток Б. Глосарій

- **Автентифікація** – ***Authentication*** – оцінка відповідності наданої ідентифікаційної інформації користувача додатка.
- **Автоматизоване оцінювання відповідності** – ***Automated Verification*** – використання автоматизованих інструментів (інструментів динамічного та / або статичного аналізу), що виявляють проблеми за допомогою сигнатур вразливостей.
- **Архітектура безпеки** – ***Security Architecture*** – абстракція проекту додатка, що визначає та описує, де і як застосовуються контролі безпеки, а також місцезнаходження та чутливість даних як додатка, так і його користувачів.
- **Атаки на відмову в обслуговуванні**, ***DoS*-атаки** – ***Denial of Service Attacks***, ***DoS Attacks*** – переповнення додатка більшою кількістю запитів, ніж він здатний обробити.
- **Безпека додатка** – ***Application Security*** – безпека на рівні додатка зосереджується на аналізі компонентів, охоплюючи прикладний рівень еталонної моделі взаємодії відкритих систем (моделі *OSI*[86]), а не, наприклад, на базовій операційній системі чи на об'єднаних мережах.
- **Безпека комунікацій** – ***Communication Security*** – захист даних додатка під час їх передачі між компонентами додатка, між клієнтами та серверами, а також між зовнішніми системами та додатком.
- **Безпека транспортного рівня, *TLS*** – ***Transport Layer Security*, *TLS*** – криптографічні протоколи, що забезпечують безпеку комунікацій в Інтернеті.
- **Білий список** – ***Whitelist*** – перелік дозволених даних або операцій, наприклад, перелік символів, дозволених при перевірці введених даних.
- **Великодні яйця** – ***Easter Eggs*** – тип зловмисного коду, який не запускається, поки користувач не введе специфічні дані.
- **Вихідне кодування** – ***Output Encoding*** – канонікалізація та перевірка вихідних даних додатка у веб-браузерах і зовнішніх системах.
- **Відкритий проект захисту веб-додатків**, **OWASP** – ***Open Web Application Security Project***, ***OWASP*** – безкоштовна відкрита міжнародна спілка, що займається вдосконаленням безпеки прикладних програм, місія якої полягає в привертанні уваги до питання безпеки додатків, щоб як люди, так і організації мали змогу приймати інформовані рішення щодо ризиків безпеки додатків (див.: http://www.owasp.org/).
- **Вставка мови структурованих запитів**, ***SQL*-ін'єкція** – ***SQL Injection***, ***SQLi*** – техніка впровадження коду, що використовується при атаці додатків, якими управляють дані, коли в точці входу підставляються зловмисні *SQL*-оператори.
- **Глобально унікальний ідентифікатор**, ***GUID*** – ***Globally Unique Identifier***, ***GUID*** – унікальний реєстраційний номер, що використовується як ідентифікатор програмного засобу.

[86] *OSI* (*Open Systems Interconnection*) – взаємодія відкритих систем.

- **Динамічна оцінка відповідності** – ***Dynamic Verification*** – використання автоматизованих інструментів, що виявляють проблеми під час виконання додатка за допомогою сигнатур вразливостей.
- **Забезпечення випадковості шарів адресного простору**, **ASLR** – ***Address Space Layout Randomization***, ***ASLR*** – техніка захисту від атак на основі переповнення буфера.
- **Закладка** – ***Back Doors*** – тип зловмисного коду, що дозволяє отримати неавторизований доступ до додатка.
- **Звіт щодо оцінки відповідності безпеки додатків** – ***Application Security Verification Report*** – звіт, у якому містяться загальні результати та допоміжний аналіз, здійснений контролером відповідності окремого додатка.
- **Зовнішні системи** – ***External Systems*** – серверний додаток або сервіс, який не є частиною додатка.
- **Інформація, що дозволяє ідентифікувати особистість**, **PII** – ***Personally Identifiable Information***, ***PII*** – інформація, яку можна використовувати як окремо, так і в поєднанні з іншою інформацією з метою ідентифікації, визначення місцезнаходження чи встановлення контакту з окремою людиною, або ідентифікації особи в контексті.
- **Каскадні таблиці стилів**, ***CSS*** – ***Cascading Style Sheets***, ***CSS*** – мова таблиць стилів, що використовується для опису семантики представлення документів, написаних мовою розмітки даних, як, наприклад, *HTML*.
- **Компонент** – ***Component*** – автономна одиниця коду з відповідним дисковим і мережевим інтерфейсами, яка взаємодіє з іншими компонентами.
- **Контролер відповідності** – ***Verifier*** – людина чи група людей, що перевіряють додаток на відповідність вимогам *ASVS OWASP*.
- **Контроль безпеки** – ***Security Control*** – функція чи її складова частина, що виконує перевірку безпеки (наприклад, перевірка контролю доступу) або визначає термін отримання запитуваних результатів у системі безпеки (наприклад, генерація записів аудиту).
- **Контроль доступу** – ***Access Control*** – засоби, що обмежують доступ до файлів, функцій посилань, *URL*-адрес і даних, які ґрунтуються на ідентифікаторах користувачів і / або груп, яким вони належать.
- **Конфігурація безпеки** – ***Security Configuration*** – динамічна конфігурація додатка, що впливає на функціонування контролів безпеки.
- **Криптографічний модуль** – ***Cryptographic Module*** – апаратне забезпечення, програмні засоби та / або вбудовані програми, що впроваджують криптографічні алгоритми та / або генерують криптографічні ключі.
- **Міжсайтове виконання сценаріїв**, ***XSS*** – ***Cross-Site Scripting***, ***XSS*** – вразливість системи безпеки, притаманна, як правило, веб-додаткам, які дозволяють введення клієнтських скриптів у свій вміст.
- **Мова розмітки гіпертекстових документів**, ***HTML*** – ***HyperText Markup Language***, ***HTML*** – основна мова розмітки для створення веб-сторінок та іншої інформації, що відображається у веб-браузері.

- **Моделювання загроз** – ***Threat Modeling*** – техніка постійного вдосконалення архітектур безпеки з метою визначення агентів загроз, зон безпеки, контролів безпеки та важливих технічних засобів і бізнес-активів.
- **Оцінка відповідності безпеки додатків** – ***Application Security Verification*** – технічне оцінювання додатків згідно з *ASVS OWASP*.
- **Оцінка відповідності проекту** – ***Design Verification*** – технічне оцінювання архітектури безпеки додатка.
- **Перевірка вхідних даних** – ***Input Validation*** – канонікалізація та перевірка даних, введених ненадійним користувачем.
- **Позитивна перевірка** – ***Positive Validation*** – див. білий список.
- **Полегшений протокол доступу до директорій**, ***LDAP*** – ***Lightweight Directory Access Protocol***, ***LDAP*** – протокол додатка для забезпечення доступу та обслуговування інформаційних служб розподілених у мережі директорій.
- **Протокол передачі гіпертекстових документів**, **HTTP** – ***HyperText Transfer Protocol***, ***HTTP*** – протокол додатка для розподілених, сумісних і гіпермедійних інформаційних систем, який є основою передачі даних у всесвітній павутині.
- **Розширювана мова розмітки**, ***XML*** – ***XML*** – мова розмітки, яка встановлює низку правил для кодування документів.
- **Стандарт *FIPS 140-2*** – ***FIPS 140-2*** – стандарт, який можна використовувати як основу для оцінювання відповідності проекту та впровадження криптографічних модулів.
- **Статична оцінка відповідності** – ***Static Verification*** – використання автоматизованих інструментів, що виявляють проблеми у вихідному коді додатка за допомогою сигнатур вразливостей.
- **Тестування прийнятності для користувача**, **ТПК** – ***User Acceptance Testing***, ***UAT*** – традиційно тестове середовище, що відповідає робочому середовищу, в якому здійснюється тестування програмного засобу до його запуску.
- **Уніфікований ідентифікатор ресурсів**, ***URI***[87] – ***URI*** – лінійка символів, що використовується для визначення назви або веб-ресурсу.
- **Уніфікований локатор ресурсів**, ***URL*** / **фрагменти *URL*** – ***URL*** / ***URL Fragments*** – лінійка символів, що часто використовується як посилання на ресурс.
- **Центр сертифікації** – ***Certificate Authority***, ***CA*** – суб'єкт, що видає цифрові сертифікати.
- **Цільовий об'єкт оцінки відповідності** – ***Target of Verification***, ***TOV*** – певний додаток, щодо якого здійснюється оцінювання відповідності безпеки згідно з вимогами *ASVS OWASP*.
- **Чорний список** – ***Blacklist*** – перелік заборонених даних або операцій, наприклад, перелік символів, які не можна вводити.
- **Шкідливий код** – ***Malicious Code*** – код, впроваджений у додаток під час його розробки без відома власника додатка в обхід діючої політики безпеки (не плутати зі зловмисним програмним засобом, як, наприклад, вірус або хробак).

[87] *URI* (*Uniform Resource Identifier*) – уніфікований ідентифікатор ресурсів.

- **Шкідливий програмний засіб** – ***Malware*** – робочий код, що впроваджується у додаток під час його виконання без відома власника додатка чи адміністратора.

Додаток B. Посилання

Наступні проекти *OWASP* з великою ймовірністю можуть бути корисними користувачам / впроваджувачам цього *Стандарту*:

- *Настанови OWASP щодо тестування* (https://www.owasp.org/index.php/OWASP_Testing_Project);
- *Настанови OWASP щодо аналізу коду* (http://www.owasp.org/index.php/Category:OWASP_Code_Review_Project);
- *Пам'ятки OWASP* (https://www.owasp.org/index.php/OWASP_Cheat_Sheet_Series);
- *Проактивні контролі OWASP* (https://www.owasp.org/index.php/OWASP_Proactive_Controls);
- *Топ-10 OWASP* (https://www.owasp.org/index.php/Top_10_2013-Top_10);
- *Мобільний топ-10 OWASP* (https://www.owasp.org/index.php/Projects/OWASP_Mobile_Security_Project_-_Top_Ten_Mobile_Risks).

Окрім того, користувачам / впроваджувачам цього Стандарту з великою ймовірністю можуть бути корисними наступні веб-сайти:

- *Перелік розповсюджених слабких сторін MITRE*[88] (http://cwe.mitre.org/);
- Рада зі стандартів безпеки даних індустрії платіжних карток (https://www.pcisecuritystandards.org):
    - *Вимоги та процедури оцінювання безпеки згідно з PCI-DSS версії 3.0* (https://www.pcisecuritystandards.org/documents/PCI_DSS_v3.pdf).

[88] *MITRE* (*The MITRE Corporation*) – американська організація, що управляє низкою дослідницьких центрів, які фінансуються урядом.

Додаток Г. Встановлення відповідності стандартам

Вимоги 6.5 *PCI-DSS* ґрунтуються на *Топ-10 OWASP 2004-2007 рр.* і додатково описують останні розширення процесів. *ASVS* є розширеним строгим варіантом *Топ-10 OWASP 2013 р.* (154 з 10 пунктів). Таким чином, усі питання, які охоплює *Топ-10 OWASP* і вимоги 6.5.*x PCI DSS*, описуються в більш детальних вимогах до контролів *ASVS*. Наприклад, «Порушена автентифікація та управління сеансами» точно відповідає Розділам ОВ2 «Автентифікація» та ОВ3 «Управління сеансами».

Повна відповідність досягається за допомогою перевірки третього рівня, хоча перевірку другого рівня можна прослідкувати до вимог 6.5 *PCI DSS*, окрім 6.5.3 та 6.5.4.

ASVS не охоплює питання щодо процесів, як, наприклад, вимогу 6.5.6 *PCI DSS*.

***PCI-DSS* версії 3.0**	***ASVS* версії 3.0**	**Опис**
6.5.1. Помилки ін'єкцій, зокрема вставки мови структурованих запитів. Додатковий розгляд помилок ін'єкцій в *OS*-команди, *LDAP* та *XPath* тощо.	5.11, 5.12, 5.13, 8.14 і 16.2.	Пряма відповідність.
6.5.2. Переповнення буфера.	5.1.	Пряма відповідність.
6.5.3. Незахищене зберігання криптографічних даних.	Весь Розділ ОВ7.	Повна відповідність з першого рівня і вище.
6.5.4. Незахищені комунікації.	Весь Розділ ОВ10.	Повна відповідність з першого рівня і вище.
6.5.5. Неправильна обробка помилок.	3.6, 7.2, 8.1 і 8.2.	Пряма відповідність.
6.5.7. Міжсайтове виконання сценаріїв (*XSS*).	5.16, 5.20, 5.21, 5.24, 5.25, 5.26, 5.27, 11.4 та 11.15.	*ASVS* розділяє *XSS* на декілька вимог, які висвітлюють комплексність захисних механізмів *XSS*, зокрема для додатків попередніх версій.
6.5.8. Неналежний контроль доступу (як, наприклад, небезпечні прямі посилання на об'єкти, неможливість обмежити *URL*-доступ, обхід каталога та неможливість обмежити доступ користувачів до функцій).	Весь Розділ ОВ4.	Повна відповідність з першого рівня і вище.
6.5.9. Підробка міжсайтових запитів (*CSRF*).	4.13.	Пряма відповідність. *ASVS* розглядає захист від *CSRF* як один із аспектів

		контролю доступу.
6.5.10. Порушена автентифікація та управління сеансами.	Розділи OB2 та OB3 цілком.	Повна відповідність з першого рівня і вище.