УДК 004

М.Дроздов, магістр гр. КІ-24М,
Центральноукраїнський національний технічний університет

ДОСЛІДЖЕННЯ ТА ПРИНЦИПИ ПОБУДОВИ СИСТЕМИ ЗАХИСТУ КОРПОРАТИВНИХ ДОДАТКІВ, ЯКІ РОЗРОБЛЯЮТЬСЯ ЗА ДОПОМОГОЮ МЕТОДОЛОГІЇ AGILE

У статті розроблено програмне забезпечення, яке призначено для системи захисту корпоративних додатків, які розробляються за допомогою методології Agile. Метою розробки є дослідження та принципи побудови системи захисту корпоративних додатків, які розробляються за допомогою методології Agile. Об'єктом дослідження є процес захисту корпоративних додатків, які розробляються за допомогою методології Agile. Предметом дослідження є методи захисту корпоративних додатків, які розробляються за допомогою методології Agile. Методи дослідження базуються на методах захисту інформації у комп'ютерних мережах, методах математичної статистики, методах розробки програмного забезпечення. Результат роботи – програмна реалізація системи захисту корпоративних додатків, які розробляються за допомогою методології Agile. В процесі роботи над програмною моделлю виконано аналіз існуючих апаратних та програмних засобів. В повній мірі описані всі компоненти розробленого програмного забезпечення.

система захисту, корпоративні додатки, Agile

Постановка проблеми. Гнучка розробка програмного забезпечення (Agile) – це тип методології, який зосереджений на основному принципі гнучкості. Методи гнучкої розробки визнають, що для отримання найкращих кінцевих продуктів у найкоротші терміни може знадобитися зміна культури або мислення поряд з іншими змінами в процесі. Незважаючи на широке впровадження з моменту першого впровадження на початку 2000-х років з Agile-маніфест, за останні п'ять років DevOps став стандартною методологією, яка використовується для створення чудових програмних продуктів. Однак сьогодні, DevSecOps – це золотий стандарту розробці безпечних додатків.

Створення гнучкого середовища розробки пропонує кілька ключових переваг. Сучасний бізнес працює швидше, ніж будь-коли раніше, тому команди розробників повинні мати можливість своєчасно реагувати на ринкові сили, водночас надаючи пріоритет ефективній доставці програми. Agile надає платформу, яка дозволяє командам швидко рухатися та створювати високоякісні продукти завдяки таким методам, як постійне тестування протягом усього процесу розробки. Пошук більш гнучких методів розробки дозволяє забезпечити безпечніший життєвий цикл розробки програм від коду до хмари. Успішне впровадження практик безпеки разом зі гнучкою розробкою програмного забезпечення дозволяє швидко розробляти безпечний код, або DevSecOps.

Аналіз останніх досліджень і публікацій. При аналізі останніх досліджень і публікацій [1-30] було виявлено певні прогалини у забезпеченні системи захисту корпоративних додатків, які розробляються за допомогою методології agile.

Мета й завдання дослідження. Метою роботи є дослідження та принципи побудови системи захисту корпоративних додатків, які розробляються за допомогою методології Agile.

Для досягнення поставленої мети визначена програма дослідження, що складається з наступних завдань:

– Огляд існуючих систем захисту корпоративних додатків, які розробляються за допомогою методології Agile.

– Дослідження системи захисту корпоративних додатків, які розробляються за допомогою методології Agile.

– Програмна реалізація системи захисту корпоративних додатків, які розробляються за допомогою методології Agile.

Об'єктом дослідження є процес захисту корпоративних додатків, які розробляються за допомогою методології Agile.

Предметом дослідження є методи захисту корпоративних додатків, які розробляються за допомогою методології Agile.

Методи дослідження базуються на методах захисту інформації у комп'ютерних мережах, методах математичної статистики, методах розробки програмного забезпечення.

Виклад основного матеріалу. Навіть якщо ви бездоганно дотримувалися всіх найкращих практик гнучкої розробки, у гнучкій розробці програмного забезпечення все одно існують ризики. Гнучка розробка програмного забезпечення довела, що вона пропонує більше переваг, ніж недоліків, тому важливо розуміти потенційні ризики безпеки щоб ви могли пом'якшити їх завчасно.

Відсутність документації процесу

Agile надає пріоритет робочим продуктам над документацією. Деякі Agile-команди доводять це до крайності та прагнуть до нульової документації, але більшість типових, добре функціонуючих Agile-команд зосереджуються на мінімізації документації та забезпеченні того, щоб документування додавало цінності.

Однак така практика створює певні ризики в розробці програмного забезпечення. Agile надає пріоритет гнучкості (що добре), але це може призвести до швидкої послідовності швидких змін, що полегшує втрату сліду за документацією процесу.

Коли ви втрачаєте контроль над документацією процесу, ви не можете гарантувати дотримання всіх протоколів безпеки, оскільки довідкової документації може не існувати, або її зберігання в узгодженому вигляді не було пріоритетом. Для забезпечення безпечної розробки додатків ідеально мати певну довідкову документацію, незалежну від продукту. У цьому випадку підвищена гнучкість, пов'язана з меншою кількістю документації, може збільшити вашу вразливість до порушень безпеки.

Щоб зменшити проблеми безпеки, пов'язані з браком документації, зосередьтеся на більш тісному впровадженні безпеки в процес розробки у формі «чемпіонів безпеки» в команді розробників або шляхом автоматизації деяких форм документації для створення історичного запису, до якого можна буде звертатися пізніше. Однією з головних проблем, з якими сьогодні стикаються керівники інформаційних систем (CISO) та їхні команди безпеки, є розуміння процесів розробки додатків, які їм доручено захистити, оскільки в організації немає документації щодо того, які системи використовуються розробниками. Зараз існують сучасні інструменти безпеки, такі як SaaS-платформа Legit Security, яка може автоматично виявляти та інвентаризувати активи SDLC – автоматично створюючи документацію, до якої команди безпеки можуть отримати доступ за потреби.

Цикл швидкого вивільнення

Гнучка методологія допомагає створити середовище для швидких релізів, але оскільки великий пріоритет надається забезпеченню цінності продукту протягом спринту, безпека часто відходить на другий план порівняно з розробкою основної функціональності продукту на ранніх етапах життєвого циклу розробки програмного забезпечення (SDLC). На жаль, це виявилося антипаттерном. Знижуючи пріоритет безпеки на ранніх етапах розробки програмного забезпечення, ви створюєте борг безпеки, який буде вирішено пізніше, коли для усунення цієї помилки безпеки буде набагато більше ресурсомістких ресурсів. Крім того, бізнес взяв на себе більшу відповідальність за ризик протягом цього часу, якщо помилка безпеки стане використаною вразливістю. Прагніть інтегрувати безпеку в гнучкі методи розробки.

Замість того, щоб просто пріоритезувати випуск коду, зосередьтеся на забезпеченні дотримання протоколів тестування вразливостей безпеки. Це може допомогти впровадити

підхід «безпека понад усе» до гнучкої розробки та уникнути потенційних ризиків для безпеки. У дусі «гнучої безпеки» докладайте всіх зусиль для автоматизації перевірок безпеки та програмної оцінки дотримання протоколів безпеки.

Поява нових загроз

Оскільки гнучка розробка зосереджена на постійному вдосконаленні та невеликих обсягах роботи, релізи можуть створювати нові ризики безпеці, яких можна було б уникнути, якби ми продумали це наперед. Крім того, гнучкі методології іноді використовуються як привід для уникнення ретельного планування та підготовки. Команди, які недалекоглядно зосереджені лише на розробці наступного нового релізу, можуть не враховувати, як може змінитися ландшафт загроз з новим релізом (тобто розробники не є експертами з безпеки). В результаті такого сценарію зловмисники можуть легше використовувати будь-які приховані вразливості в гнучкому середовищі розробки.

Одним із поширених прикладів цього в сучасній індустрії програмного забезпечення є використання жорстко закодованих секретів у вихідному коді. Ця практика не викликала великого занепокоєння десять років тому, коли концепція хмарного програмного забезпечення, що надається як послуга, перебувала на ранніх стадіях впровадження на ринку. Однак протягом останнього десятиліття більшість гнучких розробників програмного забезпечення не зробили жодного кроку назад, щоб оцінити вплив SaaS та жорстко закодованих секретів. Це створило новий вектор атаки, де зловмисники прагнуть отримати доступ до систем управління вихідним кодом та вихідного коду, щоб отримати подальший доступ до інших систем (наприклад, серверів збірки), крадучи жорстко закодовані секрети. Короткозоро зосереджуючись на кожному спринті. Програмна галузь в цілому пропустила кричущу проблему безпеки, яка тепер стала головною проблемою.

Коли виникають нові загрози, вирішуйте їх двома способами:

Аудит вашої існуючої поверхні атаки, щоб оцінити поточний ризик і визначити пріоритети виправлень безпеки.

Вбудовуйте методи для ітеративного аудиту поверхні атаки, що постійно та автоматично відповідає вашому існуючому процесу розробки, щоб запобігти подальшому ризику.

Щоб впоратися з появою нових загроз за допомогою цього двостороннього підходу, організаціям необхідно впровадити стратегію внутрішніх команд для покращення безпеки та безперервного навчання.

Недостатня обізнаність та навчання з питань безпеки

Загальновідомо, що безпека часто відходить на другий план як з боку внутрішніх команд, так і з боку клієнтів та сторонніх ресурсів на користь більш безпосередньої діяльності, що «генерує дохід». В епоху атаки на ланцюги поставок програмного забезпечення. Однак таке мислення небезпечно застаріло. Простіше кажучи, атака SolarWinds чітко показала, що існує новий тип загрози, яка безпосередньо призведе до втрати клієнтів, оскільки зловмисники тепер прагнуть атакувати ваших клієнтів *через вас.* Ось чому важливо підкреслювати та сприяти підвищенню обізнаності та навчанню з питань безпеки для всіх сторін.

Замість того, щоб зосереджуватися лише на швидкості та термінах виконання, виділіть спеціальний час для навчання з питань безпеки та протоколів безпеки. Прагніть досягти гнучку безпеку. Використовуйте сучасні інструменти безпеки ланцюга постачання програмного забезпечення, які надають агрегований «оцінку безпеки» за лінійкою продуктів та командою розробників. Таким чином, ви можете застосовувати більше навчання з безпеки до команд розробників, які цього найбільше потребують, не обтяжуючи при цьому вже безпечні команди розробників. Завдяки гейміфікації практик безпеки за допомогою оцінок безпеки за лінійкою продуктів, організації можуть перейти до фази посилення гнучкої безпеки.

Необмежений доступ до репозиторіїв вихідного коду та конвеєрів CI/CD

Нерідко трапляється просто надавати загальні дозволи та авторизації великим групам команди розробників, особливо в невеликих організаціях з менш розвиненими практиками безпеки. Більша кількість користувачів з доступом до репозиторіїв коду, систем керування вихідним кодом (SCM), серверів збірки, реєстрів артефактів та конвеєрів CI/CD означає більший ризик для вашого SDLC. Кожен користувач із надмірними привілеями являє собою ще одне слабке місце у вашому захисті, яке стає мішенню для зловмисників. Крім того, команди розробників програмного забезпечення часто передають певну частину розробки зовнішнім, контрактним командам розробників на проектній основі. Часто ці підрядники зберігають свої дозволи ще довго після завершення проекту. Ризики для гнучкої розробки програмного забезпечення особливо критичні, коли привілеї адміністративного рівня надаються занадто вільно.

Підвищте свою гнучку безпеку, дотримуючись найкращих практик гнучкої розробки. Перегляньте дозволи та протоколи авторизації для вашої команди розробників та систем, які вони використовують для швидкої розробки програмного забезпечення. Використовуйте принцип найменших привілеїв, щоб кожен користувач мав лише ті дозволи, які необхідні для виконання його роботи.

Якщо кількість розробників, які мають доступ до ваших конвеєрів розробки, коливається від 0 до 50 осіб, ручний підхід може бути можливим і бажаним; однак для команд, що налічують понад 50 розробників, найкраще використовувати такий інструмент, як платформа Legit Security, щоб допомогти вам визначити поточні ризики доступу, присутні у ваших конвеєрах розробки, перш ніж зловмисник скомпрометує застарілі облікові записи, які, як ви помилково вважали, більше не мають доступу до ваших конвеєрів.

Невміння керувати конфіденційними даними

Багато команд розробників розуміють основні принципи безпеки, але більшість із них не дуже добре обізнані з тим, як інтегрувати практики безпеки навколо управління даними таким чином, щоб це відповідало найкращим практикам гнучкої розробки.

Повний перелік ризиків безпеки, пов’язаних з управлінням конфіденційними даними, є довгим і виходить за рамки цього блогу, але деякі з найбільших викликів у DataOps – це (1) впровадження проактивного управління даними, (2) маркування конфіденційних даних та (3) навчання команд прийнятним практикам роботи з даними.

Ось кілька порад щодо кращого управління конфіденційними даними:Централізувати управління ідентифікацією;Визначення прав доступу на основі ролей;Маскування конфіденційних даних;Відмовтеся від ручного захисту даних та *автоматизуйте його*, коли це можливо;Постійно скануйте код на наявність конфіденційних даних.

Оскільки команди безпеки та розробки стають більш узгодженими, вигідно мати «чемпіона безпеки» або когось у команді розробників, хто виступає за покращення загальної ситуації та наполягає на цьому. поза безпеки та управління конфіденційними даними. Врахування безпеки як частини розробки допомагає всім командам долати перешкоди у сфері безпеки.

Погане управління сторонніми розробниками та відкритим кодом

Сторонні таз відкритим кодом. Код може відігравати важливу роль для більшості команд розробників, звільняючи багато ресурсів від необхідності створювати кожен аспект коду власними силами. Однак це не означає, що сторонній код та код з відкритим вихідним кодом не потребують підтримки. Весь цей зовнішній код поєднується з власним кодом, створюючи частину вашого ланцюжка постачання програмного забезпечення, яка є дуже вразливою до атак.

Прості способи управління програмним забезпеченням з відкритим кодом та сторонніми розробниками у вашому ланцюжку поставок включають:Встановлення політики щодо використання програмного забезпечення з відкритим вихідним кодом;Створення ради управління відкритим кодом;Створення взаємопов'язаних відносин з вашими постачальниками, щоб у разі виникнення проблем вони швидко реагували та вживали

необхідних заходів для їх усунення; Регулярно запускайте сканування за допомогою аналізу складу програмного забезпечення (SCA).

Інвентаризуйте свій SDLC, щоб зрозуміти, які конвеєри є найважливішими для бізнесу.

Скануйте свої конвеєри розробки на наявність SCA-сканувань.

Важливо зазначити, що ваш ланцюжок постачання програмного забезпечення – це не лише зовнішній код, який тут використовується. Ваш ланцюжок постачання програмного забезпечення також складається з інструментів розробника та конвеєрів у вашому SDLC, а також будь-якого додаткового коду, бібліотек та ресурсів, що використовуються під час розгортання вашого програмного забезпечення або інструменту, що передаються у ваше робоче середовище.

Невикористання стороннього коду на свою користь

Багато команд надають великого значення розробці внутрішніх рішень безпеки, але розробка власного коду та рішень має кілька недоліків. Через це іноді нове, неперевірене, незахищене внутрішнє рішення використовується *за звичкою,* замість того, щоб просто оцінити сторонні рішення, які вирішують проблему більш нестандартним та менш ресурсоємним способом порівняно з саморобним підходом.

Один гібридний підхід полягає в інвестуванні в low-code та no-code рішення для безпеки та автоматизації, щоб інтегрувати безпеку в гнучкі методи розробки. Такі інструменти допомагають усунути вузькі місця, пов'язані з обмеженнями потужностей команди розробників. Власники гнучких продуктів не завжди по-справжньому розуміють наслідки для безпеки своїх основних функцій, тому важко планувати заздалегідь під час розробки вимог. І не всі команди здатні формулювати вимоги, що стосуються проблем безпеки, не диктуючи потім занадто багато елементів рішення, що може призвести до значного розширення обсягу робіт під час спроби створення внутрішніх рішень.

Коротше кажучи, не бійтеся використовувати надійний та безпечний сторонній код і рішення на свою користь, оскільки ці рішення добре перевірені та значно скорочують час розгортання.

Нездатність визначити пріоритети служби безпеки

Підхід, що ставить безпеку на перше місце, може здаватися суперечливим темпам типової гнучкої розробки для багатьох гнучких команд. Тестування та процедури безпеки часто розглядаються як перешкоди та лежачі поліцейські на шляху до швидшої доставки готового продукту членами гнучкої команди, зосередженими на дотриманні останнього дедлайну спринту.

Один зі способів підвищити безпеку в гнучкій розробці – це спричинити культурні зміни, які зроблять безпечну розробку невід'ємним пріоритетом. Приділіть час переосмисленню процедур безпеки, щоб члени команди розуміли, чому певна практика безпеки є важливою. Інтегруйте команди безпеки та розробки, щоб безпека могла бути інтегрована на ранніх етапах процесу експертом у цій галузі. Обговоріть та встановіть розумну частоту та процедури тестування, щоб допомогти перейти до культури «гнучої безпеки». Існує тісний зв'язок між додаванням занадто великої кількості необґрунтованих перевірок безпеки в процес та часом розробки, який витрачається на винахід способів обійти складні процеси. Наприклад, може бути безпечніше вимагати понад 2 рецензентів коду під час просування коду, але це створює більш ніж удвічі більше труднощів, ніж вимога лише одного рецензента коду. Відділи безпеки повинні враховувати вплив на досвід розробника під час запровадження політики, якої необхідно дотримуватися.

Вибір самостійної безпеки

Нерідко багато команд намагаються вирішувати всі свої потреби в безпеці самостійно. Існує поширена помилкова думка, що повністю внутрішня команда може заощадити вам час і гроші (особливо враховуючи те, що компанії відчайдушно потребують експертів з кібербезпеки.) Часто внутрішнім командам бракує всебічної експертизи, необхідної для забезпечення безпеки всього SDLC – саме тут і знадобляться експерти.

Зовнішня експертиза буває двох форм:Постачальники послуг кібербезпеки.

Постачальники продуктів кібербезпеки; Використання зовнішньої експертизи з безпеки для управління вашою безпекою дає деякі помітні переваги: Звільняє внутрішні ресурси, щоб зосередитися безпосередньо на завданнях, які можуть виконати тільки вони; Заповнює прогалини в спеціалізації, для забезпечення яких було б надто важко найняти внутрішній ресурс.

Тепер, коли розробка дедалі більше кодифікується, у індустрії програмного забезпечення спостерігається поява принципу «політика як код». SaaS-рішення, такі як Легітимна платформа безпеки. Зараз існують стандарти, які кодифікували сотні найкращих практик у вузькоспеціалізованих сферах безпеки, таких як безпека ланцюга постачання програмного забезпечення. Замість того, щоб вимагати внутрішнього експерта з предметної області, ви можете використовувати знання, кодифіковані в продукті. Крім того, SaaS краще підходить для контролю того, чи порушують команди розробників необхідні політики, порівняно з внутрішнім ресурсом безпеки. Зрештою, це допомагає досягти більшої безпеки-орієнтованої розробки.

Гнучка розробка допомагає компаніям-розробникам програмного забезпечення впроваджувати легкі, ітеративні цикли розробки. Методологія розробки робить акцент на невеликих, керованих командах з міжфункціональною співпрацею для частих оновлень та випусків.

Хоча Agile залежить від масштабних культурних змін у розробці та тестуванні коду, ця модель також вимагає нового підходу до впровадження безпеки в Agile.

У цьому розділі обговорюються відповідні ризики, що виникають внаслідок неправильного впровадження методів безпеки, а також заглиблюються в найкращі практики забезпечення циклу Agile-розробки.

Ризики неврахування безпеки у вашій гнучкій розробці

Гнучкі процеси зазвичай відкривають виробниче та розробницьке середовища для розподілених команд, що призводить до витоку важливої інформації через поступове збільшення кількості авторизацій. Хоча гнучкі методології розвивалися, щоб враховувати інновації та терміни виконання, безпека часто відходить на другий план, оскільки оцінки безпеки уповільнюють розробку.

Практики безпеки особливо складно впроваджувати однаково протягом усього життєвого циклу розробки програмного забезпечення. Цикл збірки є безперервним, і тести безпеки необхідно проводити поза робочим процесом розробки. Це створює різні ризики для безпеки організації, зокрема:

Недостатня політика, що регулює використання компонентів з відкритим кодом

Основний принцип Agile полягає в тому, щоб не винаходити велосипед. Крім того, командам рекомендується використовувати рішення з відкритим кодом та комерційні власні рішення для скорочення життєвого циклу розробки. Однак, хоча такі інтеграції з відкритим кодом пропонують різні переваги, вони також створюють значні ризики безпеки, які можуть вплинути на розробку.

Типовий сценарій – це коли розробники не знають про залежності модулів та кількість бібліотек коду, які вони імпортують. Як наслідок, відстеження змін у відкритому коді є складним завданням, що ускладнює забезпечення надійної та безпечної роботи програмного забезпечення.

Залежність від інтеграцій з відкритим кодом також ускладнює отримання інформації про конкретні вразливості та виправлення, виявлені спільнотою учасників. За відсутності чітко визначених політик, що регулюють інтеграції з відкритим кодом, вибір інструментів та управління технологічним стеком, виявляють проблеми для зменшення загроз.

Необмежений доступ до процесів розробки та репозиторіїв вихідного коду

Agile сприяє ефективній співпраці між міжфункціональними, розподіленими командами, які дотримуються ітеративного підходу до розробки. Щоб забезпечити роботу

команд з єдиним джерелом достовірної інформації, організації покладаються на публічні та приватні репозиторії вихідного коду для підтримки, обміну та версіонування коду.

Хоча репозиторії забезпечують безперебійний контроль версій, команди часто не можуть інтерпретувати версію вихідного коду, що використовується, що ускладнює виявлення конкретних вразливостей та застосування заходів щодо їх усунення.

Крім того, коли приватне створення коду потрапляє до репозиторіїв вихідного коду, він розкриває вразливі точки входу. Зловмисники націлюються на такі відкриті репозиторії, щоб виявити приховані функції, а потім поєднують їх з розвідувальними тактиками для організації атак.

Небезпечне управління конфіденційними даними

Сучасний процес гнучкої розробки інтегрує численні джерела для аналітики та операцій з даними в масштабах всієї організації. Складна інтеграція вимагає систематичної та точної класифікації даних для контролю рівня доступу до конфіденційних даних організації.

Крім того, фрагменти тестових даних, створені командою тестувальників, часто є значною часткою реальних даних. Хоча такі дані використовуються внутрішньо для цілей тестування, відсутність належної санітарної обробки даних сприяє зловживанню вразливостями з боку векторів атак.

Більшість Agile-організацій не мають політик захисту своїх даних від внутрішніх зловмисників. Через відсутність суворих політик класифікації даних, члени команди зазвичай не знають про комплексні вимоги безпеки, чутливість даних та свою відповідальність за їх захист. Рівень співпраці, необхідний для безперебійної роботи, також ускладнює для команди безпеки впровадження заходів безпеки.

Методології безпеки в Agile-розробці

Гнучка розробка програмного забезпечення запроваджує культурні зміни в масштабах усієї організації, зосереджуючись на модульних, більш керованих, міжфункціональних командах. Хоча ці зміни забезпечують дуже гнучку розробку, яка реагує на мінливі вимоги, організаціям потрібне впровадження спеціальних методів для захисту коду та середовищ розгортання.

Нижче наведено деякі з найкращих практик для захисту застосунків у гнучкому середовищі:

Захист гнучких конвеєрів доставки

Методологія Agile робить акцент на конвеєрах безперервної інтеграції (CI) та безперервної доставки (CD), щоб забезпечити швидше розгортання. Щоб мінімізувати ризики безпеки, команди Agile повинні розглянути інтеграцію автоматизованих механізмів тестування на кожному етапі SDLC. Крім того, всі коміти, зроблені командою розробників, повинні проходити через експертів з безпеки, щоб гарантувати безпеку розробленого застосунку.

Окрім ручних перевірок коду, також повинні бути статичні аналізатори коду, інтегровані з конвеєром неперервної інтеграції (CI), та автоматизовані модульні тести для запуску запланованих перевірок. Також рекомендується проводити активні аудити безпеки та автоматичні перевірки безпеки під час процесу CI/CD, щоб допомогти виявити вразливості безпеки, перш ніж код перейде у продакшн.

Вартість виявлення вразливості безпеки на ранній стадії значно менша порівняно з пошуком її у виробництві, що може призвести до серйозних наслідків, включаючи крадіжку конфіденційних даних, неякісну продуктивність програм та втрату репутації.

Забезпечити безперервну участь усієї організації

Порівняно з традиційними практиками, Agile-процес представляє собою зсув у практиках розробки, який повинен супроводжуватися змінами в організаційній культурі в бік практик безпеки. Рекомендується, щоб фахівці з безпеки співпрацювали з розробниками для формування затвердженого керівництва щодо стандартів кодування, шаблонів проектування та рішень щодо проектування.

Організаціям також слід проводити ретельне моделювання загроз, одночасно розробляючи безперервне навчання з безпеки, яке зосереджується на зміні поведінки, а не лише на обізнаності. Фахівці з безпеки також повинні документувати критерії безпечної розробки на спільних платформах, щоб інші організації могли посилатися на них у своїх процесах.

Фірми також можуть впроваджувати гнучкі методи безпеки, впроваджуючи політики безпеки в нещодавно розроблені мікросервіси, додатки, інтеграції та API.

Впроваджуйте пасивні огляди

Методологія гнучкої розробки характеризується коротшими ітераціями, швидкими збірками та частими випусками нових функцій. Тому необхідно переглянути всі зміни в додатку, які потребують ітерацій, та як вони впливають на безпеку системи, перш ніж їх впроваджувати.

Пасивний огляд коду допомагає фахівцям з контролю якості розуміти додатки, не скануючи кожен рядок вихідного коду. Натомість, фахівці з контролю якості можуть переглядати код, а системи огляду коду відстежують їхні переміщення в кодовій базі та пропонують своєчасну, відповідну додаткову інформацію для вихідного коду. Деякі інструменти безпеки пасивного огляду також генерують схеми моделей, які візуалізують логіку, реалізовану кодом, спрощуючи усунення несправностей.

Забезпечення проактивного контролю для команд Agile-розробників

Організація повинна підтримувати розробників у впровадженні заходів контролю, що сприяють безпечній практиці розробки коду. OWASP перераховує різні методи безпеки, які забезпечують побудову всіх рівнів програми, з акцентом на безпеці. Деякі з цих методів включають:

Визначення вимог безпеки – Гнучкі методи повинні бути побудовані на галузевих стандартах, минулих вразливостях та чинному законодавстві, щоб гарантувати, що продукт відповідає всім властивостям безпеки.

Використання бібліотек та фреймворків безпеки – ці бібліотеки та фреймворки мають вбудовані заходи безпеки для захисту коду від недоліків проектування та реалізації, які призводять до недоліків безпеки.

Безпека сховищ даних – команди повинні забезпечити безпеку як NoSQL, так і реляційних баз даних за допомогою безпечних запитів, автентифікації, налаштування та зв’язку.

Кодування та екранування даних – Кодування та екранування даних використовуються для запобігання вразливостям, пов’язаним з ін’єкціями. Ескейпінг передбачає використання спеціальних символів, щоб уникнути неправильної інтерпретації рядків, тоді як кодування стосується перетворення символів на однакові значення з різними форматами, що робить їх недоступними для зловмисних інтерпретаторів.

Інші проактивні засоби контролю для комплексної Agile-безпеки включають перевірку вхідних даних, керування ідентифікацією та доступом (IAM), обробку помилок та винятків, а також ведення журналу та моніторингу безпеки,

Хоча Agile та традиційна методи безпеки спираються на впровадження подібних політик безпеки, реалізація функцій безпеки значно відрізняється. У Agile-методах швидкі цикли випуску передбачають часті тести безпеки, що означає, що детальний аналіз коду може зрештою уповільнити розробку та реалізацію.

Щоб вирішити цю проблему, індикатори ризиків безпеки слід встановити відповідно до швидкості виконання, що дозволить Agile-командам проводити перевірки безпеки, одночасно справляючись з частими змінами. Agile-практики безпеки також слід класифікувати на ті, що виконуються на початку розробки, та ті, що впроваджуються під час кожного спринту, залежно від того, чи потрібно їх виконувати одноразово чи постійно.

Найпоширенішою помилкою Agile-команд, які намагаються захистити конфіденційну інформацію, є неправильна класифікація даних. Більшість команд не вказують дані, які

потребують спеціального захисту, що ускладнює впровадження ефективної політики класифікації даних.

Крім того, команди розробників програмного забезпечення не можуть оптимально шифрувати конфіденційні дані під час передачі або зберігання, що робить їх легкодоступними для зловмисників. Ще один момент для занепокоєння виникає, коли команди неправильно використовують платформи хмарного зберігання даних, зберігаючи облікові дані та дані конфігурації без правильного тлумачення політик безпеки постачальника. Це ускладнює впровадження комплексної стратегії з точки зору безпеки.

Розроблене програмне забезпечення дозволяє забезпечити захист переданої по мережі інформації, строгу взаємну автентифікацію користувачів і серверів, гнучке розмежування доступу. Для реалізації цих функцій у системі використовуються SSL/TLS протоколи й X.509 цифрові сертифікати, тобто універсальні, що стали стандартом де-факто, механізми, підтримувані практично всіма розповсюдженими Веб-агентами.

За допомогою розробленого програмного забезпечення легко забезпечуються вимоги по інформаційній безпеці, запропоновані різними Інтернет додатками, такими як:

– сервера платіжних систем;
– інтернет-магазини;
– багатопрофільні корпоративні Веб-сервера, що містять інформацію з різним рівнем конфіденційності;
– B2B системи;
– системи захищеного документообігу;
– системи обміну електронною поштою;
– й багато які інші.

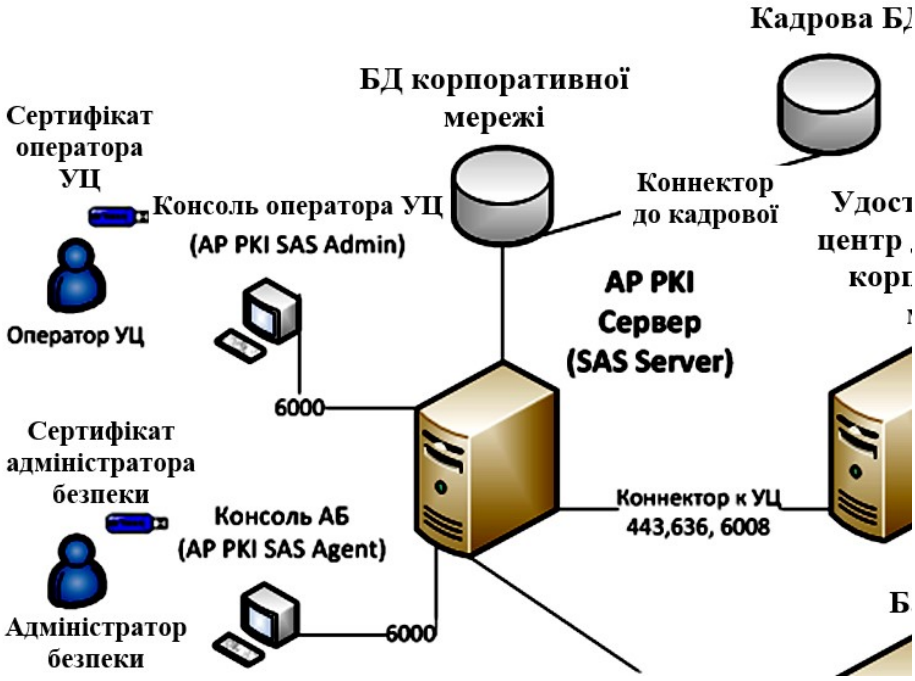

Рисунок 1 – Структурна схема системи

На рисунку 1 представлена структурна схема розробленої системи. Для поняття роботи системи введені наступні позначення:

– ЕЦП – електроний цифровий підпис;
– УЦ – удостовірюючий центр.
– ЦС – цифровий сертифікат;
– PKI – інфраструктура відкритих ключів;
– DVCS – Data Validation and Certification Server Protocols – протокол підтвердження даних та сертифікації серверу;

– OCSP – Online Certificate Status Protocol – онлайн протокол статусу сертфікату;

– TSP – Time-Stamp Protocol – протокол часових міток;

– TLS – криптографічний протокол, що забезпечує захищену передачу даних між вузлами в мережі Інтернет;

– RFC – документ, у якому описується той або інший стандарт.

Висновки. У статті наведені теоретичне узагальнення й рішення наукового завдання дослідження методів захисту корпоративних додатків, які розробляються за допомогою методології Agile. Рішення даного завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем захисту корпоративних додатків, які розробляються за допомогою методології Agile.

– Досліджена система захисту корпоративних додатків, які розробляються за допомогою методології Agile.

– На основі отриманих результатів досліджень створена програмна реалізація системи захисту корпоративних додатків, які розробляються за допомогою методології Agile. Розроблені алгоритми дозволяють успішно вирішувати завдання захисту корпоративних додатків, які розробляються за допомогою методології Agile. Проведено аналіз предметної галузі в ході якого були виявлені об'єкти, взаємодія яких носить істотний характер для функціональної діяльності предметної галузі, і їхні основні характеристики; побудована алгоритм і вибраний середовище розробки.

Список літератури

1. Kuznetsov O., Frontoni E., Kuznetsova Y., Chevardin V., Smirnov O. «Architectural foundations for adaptive security in edge computing systems». Cybersecurity Defensive Walls in Edge Computing, 2025. pp. 21-61.
2. Вінтенко, Б.Ю., Миронець, І.В., Смірнов, О.А., Коваленко, О.В., Усік, П.С., Буравченко, К.О., Лисенко, І.А. «Логіко-структурна модель комп'ютерно-орієнтованої процедури системи підтримки оперативного персоналу АЕС». Кібербезпека: освіта, наука, техніка. 2025. Том 2 № 30. С. 413-427, 2025.
3. Смірнова, Т.В. «Дослідження методів, моделей та сучасних ІТ-рішень для підтримки технологічних процесів у критичній інфраструктурі держави». Кібербезпека: освіта, наука, техніка. 2025. Том 2 № 30. С.195-208, 2025.
4. Вінтенко Б., Смірнов О., Миронець І., Смірнова Т., Смірнов С. «Імітаційна модель шляхів вхідних даних комп’ютерної інтелектуальної системи підтримки оператора енергоблоку АЕС». Комбінаторні конфігурації та їхні застосування: Матеріали XXVII Міжнародного науково-практичного семінару, присвяченого 125-річчю Національного університету «Запорізька політехніка» (Запоріжжя-Кропивницький-Київ, 4-6 червня 2025 р.). Запоріжжя: НУ «Запорізька політехніка», 2025. С.82-91.
5. Al-Azzeh, J., Ayyoub, B., Mesleh, A., Smirnova, T., Gnatyuk, S., Drieiev, O., Smirnov, O., Dorenskyi, O. «Cloud-Based Information System for Evaluating Caverns in the Process of Blasting Metal Surfaces of Details». International Review on Modelling and Simulations 18 (1), 2025. pp. 32-42.
6. Вінтенко Б.Ю., Смірнов О.А., Миронець І.В., Смірнова Т.В., Коваленко О.В., Мацуй А.М. «Модель шляхів отримання вхідних даних комп’ютерної інтелектуальної системи підтримки оперативного персоналу АЕС». Центральноукраїнський науковий вісник. Технічні науки. 2025. Вип. 11(42), ч. ІІ. С.52-62.
7. Вінтенко Б.Ю., Смірнов О.А., Миронець І.В., Смірнова Т.В. «Методи забезпечення відмовостійкості інтелектуальних систем підтримки оператора». VIII міжнародна науково-практична конференція “Інформаційна безпека та комп’ютерні технології”, м. Кропивницький. 24-25 квітня 2025 р. – Кропивницький: ЦНТУ. – 2025. – С. 44-46.
8. Смірнов, О.А., Константинова, Л.В., Конопліцька-Слободенюк, О.К., Козірова, Н.В, Якименко, Н.М., Доренський, О.П., Буравченко, К.О. «Дослідження інструментів штучного інтелекту для роботи з базами даних та аналізу даних». Кібербезпека: освіта, наука, техніка. 2025. №3(27), С. 429–448.)
9. Smirnov O., Fedorov E., Neskorodieva A., Neskorodieva T. «Intellectual Classification method of Gymnastic Elements Based on Combinations of Descriptive and Generative Approache». CEUR Workshop Proceedings Volume 3664, 2024, Pages 11-23.
10. Вінтенко, Б., Миронець, І., Смірнов, О., Коваленко, А., Конопліцька-Слободенюк, О., Смірнова, Т., Константинова, Л. «Дослідження застосування систем підтримки оперативного персоналу об’єкту критичної інфраструктури при керуванні енергоблоком АЕС з реактором типу ВВЕР-1000». Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка», 2024. № 2(26), С. 6-26.
11. Вінтенко, Б., Миронець, І., Смірнов, О., Кравчук, О., Козірова, Н., Савеленко, Г., Коваленко, А. «Дослідження вимог та аналіз кібербезпеки програмного забезпечення інформаційно-керуючих систем АЕС, важливих для безпеки». Кібербезпека: освіта, наука, техніка. 2024. №3(23), С. 111-131.
12. Kuznetsov O., Ilchenko O., Kryvinska N., Buravchenko K., Smirnov O., Savchenko Iu. «An Empirical Assessment of Leading Blockchain Financial Services». 2023 IEEE 1st Ukrainian Distributed Ledger Technology Forum

(UADLTF), Kyiv, Ukraine, 2023, pp. 1-6,
13. Kuznetsov, O., Kryvinska, N., Ilchenko, O., Smirnova, T., Ulianovska, Y. «Comparative Analysis of Cryptocurrency Trading Platforms Using the Analytic Hierarchy Process». CEUR Workshop Proceedings, 2023, 3628, pp. 106-115.
14. Malyukov V., Bebeshko B., Lakhno V., Smirnov O., Malyukova I., Mohylnyi H. «Managing the Purchase-Sale Process of Digital Currencies Under Fuzzy Conditions». Lecture Notes in Networks and Systems, 2023, 729 LNNS, pp. 104–112.
15. Al-Mudhafar Aqeel, A.M., Smirnova, T., Buravchenko, K., Smirnov, O. «The method of assessing and improving the user experience of subscribers in software-configured networks based on the use of machine learning». Advanced Information Systems, 2023, 7(2), pp. 49-56.
16. Smirnov, O., Sydorenko, V., Aleksander, M., Zhyharevych, O., Yenchev, S. «Simulation of the cloud IoT-based monitoring system for critical infrastructures». CEUR Workshop Proceedings, Volume 3530, 2023, pp. 256-265.
17. Smirnov, O., Odarchenko, R., Smirnova, T., Bondar, S., Volosheniuk, D. «Optimal Structure Construction of Private 5G Network for the Needs of Enterprises». Lecture Notes on Data Engineering and Communications Technologies, 2023, 178, pp. 208–223.
18. Вінтенко Б.Ю., Смірнов О.А., Коваленко А.С., Смірнов С.А., Буравченко К.О. «Дослідження вимог міжнародних стандартів IEC60880 та IEC62138 з розробки програмного забезпечення інформаційно-керуючих систем АЕС, важливих для безпеки». Системи управління, навігації та зв'язку, 2023, вип. 3(73), С. 155-166.
19. Вінтенко, Б., Миронець, І., Смірнов, О., Кравчук, О., Козірова, Н., Савеленко, Г., Коваленко, А. «Дослідження вимог та аналіз кібербезпеки програмного забезпечення інформаційно-керуючих систем АЕС, важливих для безпеки». Кібербезпека: освіта, наука, техніка. 2024. №3(23), С. 111-131.
20. Вінтенко Б.Ю., Смірнов О.А., Коваленко О.В., Смірнов С.А., Коваленко А.С. «Дослідження нормативних документів та галузевих стандартів розробки програмного забезпечення комп'ютерних систем управління АЕС, важливих для безпеки». Системи управління, навігації та зв'язку, 2023, вип. 2(72), С. 170-178.
21. Аль-Мудхафар Акіл Абдулхуссейн М., Смірнова Т.В., Буравченко К.О., Смірнов О.А. «Метод оцінки та підвищення користувальницького досвіду абонентів в програмно-конфігурованих мережах на основі використання машинного навчання». Сучасні інформаційні системи, 2023, том 7, № 2, С. 49-56.
22. Вінтенко Б.Ю., Смірнов О.А., Коваленко О.В., Смірнов С.А. «Дослідження нормативної документації та стандартів розробки програмного забезпечення комп'ютерних систем управління АЕС, важливих для безпеки». VI міжнародна науково-практична конференція "Інформаційна безпека та комп'ютерні технології", м. Кропивницький. 20-21 квітня 2023 р. – Кропивницький: ЦНТУ. – 2023. – С. 35-36.
23. Smirnov, O., Karapetyan, A., Fedorov, E., «Creating Neural Network and Single Solution Human-Based Metaheuristic Methods of Solving the Traveling Salesman Problem». CEUR Workshop Proceedings, Volume 3312, 2022, pp. 47-58.
24. Smirnov O., Kuznetsov A., Kryvinska N., Kiian A., Kuznetsova K. «Full Non-Binary Constant-Weight Codes». SN Computer Science, Vol 2, 337, 2021. https://doi.org/10.1007/s42979-021-00739-w.
25. Smirnov O., Kovalenko O., Kovalenko A., Kavun S. «Quantitative Risk Assessment Method Development in the Context of the SDLC-model». 2021 IEEE 8th International Conference on Problems of Infocommunications, Science and Technology (PIC S&T), 2021, pp. 203-208, doi: 10.1109/PICST54195.2021.9772143
26. Smirnova T., Gnatyuk S., Berdibayev R., Avkurova Zh., Iavich M. «Cloud-Based Cyber Incidents Response System and Software Tools». Communications in Computer and Information Science, 2021, vol 1486. Springer, Cham. pp 169-184.
27. Smirnov, O., Kuznetsov, A., Potii, O., Poluyanenko, N., Stelnyk, I., Mialkovsky, D. «Combining and filtering functions in the framework of nonlinear-feedback shift register». International Journal of Computing; 2020, Volume 19, Issue 2 – Research Institute for Intelligent Computer Systems – 2020. – P. 247-256.
28. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova T. «Non-binary constant weight coding technique». CEUR Workshop Proceedings. Volume 2740, 2020, Pages 102-114.
29. Smirnov O., Kuznetsov A., Kiian A., Cherep A., Kanabekova M., Chepurko I. «Testing of code-based pseudorandom number generators for post-quantum application». 2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT), Ukraine, Kyiv, May 14-18. 2020. P. 172-177.
30. Smirnov, O., Shekhanin, K., Kuznetsov, A., Krasnobayev, V. «Detecting Hidden Information in FAT». International Journal of Computer Network and Information Security (IJCNIS). Vol. 12, No. 3, 2020. PP.33-43.
31. Smirnov, O., Drieieva, H., Drieiev, O., Simakhin, V., Bondar, S., Odarchenko, R. «Managing multifractal properties of the binary sequence generated with the Markov chains», CEUR Workshop Proceedings Volume 2608, 2020, Pages 633-645.
32. Smirnov O. Kuznetsov A., Zaicenko Yu., Pastukhov M., Oleshko O., Kuznetsova K., «Formation of Discrete Signals with Special Correlation Properties». International Conference on Information and Telecommunication Technologies and Radio Electronics, UkrMiCo 2019; Odessa; Ukraine; 9-13 September 2019. P.22-28.
33. Smirnov, O., Kuznetsov, A., Kolovanova, I., Kuznetsova, T., «Noise immunity of the algebraic geometric codes». International Journal of Computing; 2019, Volume 18, Issue 4 – Research Institute for Intelligent Computer Systems – 2019. – P. 393-407.