CELF: ВЕБ-ПЛАТФОРМА ДЛЯ АВТОМАТИЧНОГО ВИКОНАННЯ ПОЛІТИКИ КОНФІДЕНЦІЙНОСТІ

Безрук В.М., Кривенко С.А.
Кафедра інформаційно- мережної інженерії, Харківський національний університет радіоелектроніки, Харків, Україна, E-mail: Stanislav.Kryvenko@nure.ua

Анотація - Оскільки люди розділяють більше персональних даних в Інтернеті, все більш важливо правильно застосовувати політики щодо конфіденційних даних. Щоб вирішити цю проблему, ми розробили Celf, веб-фреймворк, що дозволяє програмісту відокремити реалізацію політики потоку інформації від решти функціональних можливостей. Веб-платформа застосовує бібліотеку мови програмування Jeeves для автоматичного застосування політики конфіденційності. Наш підхід є новим у тому, що він надає наскрізні гарантії опосередковуючи взаємодії між інтерфейсом, шарами додатків і баз даних. Програмісту потрібно лише визначити політику інформаційного потоку для автоматичного впровадження через веб-платформи.

Ключові слова - веб-платформа, політики конфіденційності, Bottle, GPS, Jeeves, Python, SQLite, WSGI

I. Вступ

Оскільки люди вкладають більшу частину свого життя в Інтернет, програмісти знаходяться під дедалі більшим тиском забезпечити, щоб веб-програми правильно реалізовували очікувану політику конфіденційності. Виконання цих політик у веб-додатках може бути особливо важким, оскільки програміст повинен обґрунтовувати, як політики та дані взаємодіють через глобальні шари інтерфейсу, програми та бази даних.

Наприклад, розглянемо додаток, в якому беруть участь соціально спільні місця розташування користувачів. Навіть дотримання, здавалося б, простої політики, що точні GPS координати користувача повинні надходити лише до "друзів", нетривіально. У таких системах користувачі часто мають доступ до чутливих значень не тільки безпосередньо, наприклад, переглядаючи профіль користувача, але й побічно, наприклад, виконуючи запити до даних. На сайті соціальної мережі Facebook користувачі можуть позначати повідомлення та фотографії координатами місць, які можуть потім мати інші користувачі та здійснювати пошук за допомогою функції Graph Search. Застосування політики стає ще складнішим коли результати цих пошуків можуть бути повторно передані.

Попередні роботи представили Jeeves [1, 2], підхід до пом'якшення навантаження програміста на впровадження політики на базі мови. Мова Jeeves дозволяє програмісту окремо реалізувати інформаційні політики з інших функціональних можливостей, покладаючись на середовище виконання для отримання відповідних результатів. У Jeeves програміст визначає різні класи для кожної чутливої величини, а також політику, яка встановлює, які класи дозволено пропускати до яких контекстів. Наприклад, у задачі розташування, програміст може визначити клас високої конфіденційності, тобто точне розташування GPS, клас з низькою конфіденційністю, що показує країну, а також політику, згідно з якою клас з високою конфіденційністю може надходити лише до "друзів".

Інша програма бібліотеки Jeeves може бути агностичною політикою: програмісту потрібно лише знати, що через програму можуть проходити різні погляди, але не політика, що визначає, коли може бути показаний вигляд. Jeeves має грановану модель виконання, що імітує одночасне виконання на переглядах і використовує політику для визначення того, який результат показувати.

II. Виконання політики конфіденційності

У цьому розділі ми обговорюємо питання, пов'язані з безпекою та вибором мови та веб-платформи.

A. Веб-платформа

Пропонується Celf, веб-платформа на основі Python, що застосовує модель політики агностичного програмування для основи веб-розробки. Celf звертається до складності обробки веб-додатків, які охоплюють декілька режимів виконання і мають постійний стан. У Celf програміст пов'язує політики та перегляди з низькою конфіденційністю з даними один раз, як частина схем даних. Ця система гарантує, що політики є задоволені наскрізною поведінкою системи. Celf робить більш ніж тільки перевіряє помилки: фреймворк пристосовує поведінку програми до отримання виходів у відповідності до дозволу глядача.

Наша реалізація використовує мета-програмування Python, можливості Python з динамічного перетворення джерел, так що він виконується відповідно до гранітної семантики Jeeves, щоб забезпечити збереження гарантій навіть за наявності стійкого стану. Ми розширили веб-фреймворк Bottle Python, щоб реалізувати аспект-обізнаність шару об'єктно-реляційного відображення (ORM). Celf ORM опосередковує доступ до бази даних і гарантує, що конфіденційність гарантується навіть у сесіях. Використовуючи ці механізми, Celf надає переваги програмно-агностичного програмування в контексті знайомої моделі програмування: програми Celf виглядають як програми Bottle, де схеми даних мають додаткові декларації щодо політики інформаційного потоку. Celf не вимагає розширення інтерпретатору Python і працює зі стандартною базою даних SQLite.

B. Обговорення безпеки

Метою Jeeves є запобігання програмістам ненавмисного витоку політики. Самі програми довіряють інтерпретатору Python і базі даних, а також програмісту, щоб правильно вказати політику. У програмі Celf є лише два місця, де потрібно довіряти коду. По-перше, самі схеми та визначення політики. Звичайно, якщо політика оголошена неправильною, то можуть виникнути витоки конфіденційності. Інше місце - коли дані спочатку вводяться користувачем, і програма спочатку поміщає дані в модель і зберігає її. До тих пір, поки програміст не збереже дані, дані будуть плавати навколо вільно, і програміст міг мати витік їх в цій точці. Поза межами цих областей коду немає помилки "перевірка відсутності доступу" для перегляду даних.

Ми не захищаємо від шкідливого програміста. Зловмисний програміст міг би, наприклад, отримати доступ до "прихованих" полів граней для вилучення приватних значень. (У Python немає таких речей, як приватне поле.) Через гнучку і динамічну природу Python ми вважаємо, що було б дуже важко видалити всі подібні вразливості. Тим не менш, Jeeves має намір захищатися від недбалого програміста, і програмісту доведеться діяти дуже навмисно, щоб знищити час виконання Jeeves. Необхідно легко перевіряти такі напади.

C. Рішення про використання Python

Ми розглянули реалізацію Celf у Scala, мові, для якої вже існує реалізація Jeeves [1], але ми вибрали Python, оскільки динамічний підхід Jeeves більше підходить до ідіом динамічного програмування Python. Політично-агностичний

підхід Jeeves поширює накладні витрати на розробників на накладні витрати, що робить його більш придатним для швидкого створення прототипів і для створення веб-додатків у менших масштабах. Незважаючи на те, що Scala набуває все більшого потягу в промисловості і використовується в таких компаніях, як Twitter і Linkedln, вона не була мовою вибору для випадкового програміста, який створював рекреаційний веб-додаток.

Найбільша відмінність у підході до вбудовування мови в Python полягає в тому, що в той час як Scala є статично типізованою, мова Python динамічного вводу. Статичні типи Scala змушують програміста думати більш ретельно і більш детально в реалізації. Компроміс полягає в тому, що статичні типи забезпечують більше гарантій щодо випадків, які охоплює реалізація, що робить його більш важливим для ретельного тестування коду. Як Python, так і Scala дозволяють перевантажувати більшість, але не всі оператори. З Scala, робота навколо полягає у використанні розширень компіляторів, таких як Scala-Virtualized [3], які дозволяють програмістам перевантажувати конструкції ключів. З Python ми змогли обійти відсутність повного перевантаження, виконавши перетворення джерела. У Python це було просто, тому що абстрактне синтаксичне дерево є відносно простим, і бібліотека Python MacroPy [4] для реалізації макросів забезпечує підтримку.

D. Рішення про використання Bottle

Наша веб-платформа наразі є спеціальним шаром поверх Bottle. Ми вирішили використовувати Bottle, тому що це добре відомий веб-фреймворк з підтримкою різноманітних функціональних можливостей. Виявилося, що нам потрібно було налаштувати практично всю функціональність Bottle, яку ми використовували. Таким чином, більш легкий каркас може бути більш оптимальним вибором для базової веб-платформи.

III. Висновки

Щоб зрозуміти наведений тут зміст, немає необхідності мати базові знання про WSGI, оскільки Bottle намагається у будь-якому випадку утримувати WSGI від користувача. Але необхідно добре розуміти мову програмування Python. Крім того, веб-платформа, витягує та зберігає дані в базі даних SQLite, тому глибокі знання мови SQL допомагають, але не є обов'язковим для розуміння понять Bottle. Вихід Bottle, надісланий браузеру, відформатований в деяких прикладах за допомогою HTML. Таким чином, основи знаннь загальних HTML-тегів також допомагають.

IV. Список літератури

[1] Thomas H. Austin and Cormac Flanagan. Multiple facets for dynamic information flow. SIGPLAN Not., 2012.

[2] Jean Yang, Kuat Yessenov, and Armando Solar-Lezama. A language for automatically enforcing privacy policies. POPL, 2012.

[3] Adriaan Moors, Tiark Rompf, Philipp Haller, and Martin Odersky. Scalavirtualized. In Proceedings of the ACM SIGPLAN 2012 Workshop on Partial Evaluation and Program Manipulation, PEPM '12, New York, NY, USA, 2012. ACM.

[4] MacroPy. https: //github. com/lihaoyi/macropy.