テストテンプレートを用いたセキュリティ設計パターンの実装の適用検証

芳澤正敏[†1]　鷲崎弘宜[†2]　深澤良彰[†3]

セキュリティパターンはソフトウェア開発のためのセキュリティ専門家の知識を体系化したものだが、これらのパターンを用いる開発者はセキュリティ専門家ではないためにパターンが適切に適用されない場合がある。そこで本研究では実装段階でのセキュリティパターンの適用検証のための手法を提案する。本手法はセキュリティパターンから作成したテストテンプレートに設計情報を与えることによって設計のセキュリティパターンが妥当であるどうかの検証のためのテストを作成し検証する。

1. はじめに

セキュリティの関心事はソフトウェア開発の要求から設計、実装、テスト、運用の全てのフェーズで扱えるようにしなくてはならない [1]。しかしながら、ソフトウェア開発者が必ずしもセキュリティの専門家ではなく、開発の早い段階でのシステムにおけるセキュリティ項目について十分に考慮されていない。そこでセキュリティの専門家の知識を活用・再利用する方法としてセキュリティパターン [2]がある。セキュリティパターンとは、セキュリティにおける繰り返し起こりうる問題に対する解決策をまとめたものである。

しかし、現状では設計レベルのパターンは記述が抽象的なため、設計方針は理解できても、それを具体的にどう実装に落としていけばよいかが分かりにくい [3]。

そこで本稿では、テストテンプレートを用いたテストによるセキュリティ設計パターンの適用が妥当であるかどうかの検証を提案する。

2. セキュリティパターン

セキュリティ機能は、できる限り安全性が確認されている既存のにものを利用し、独自に実装しないことが望ましい。そのためにセキュリティ機能をどのように使えばよいのかの設計方針を示したセキュリティ設計パターンの利用がある。M.Schumacherらは[2]で、セキュリティの設計に役立つ 7 種類、38 パターンが紹介されている。

セキュリティ設計パターンの代表例として、組織や役割ごとに情報のアクセス制御について説明しているロールベースアクセス制御パターン(Role-Based Access Control Pattern)が挙げられる。このパターンでは、図 1 のようにロールベースアクセス制御を行う際のユーザと情報との静的な構造をクラス図によって示している。しかし、このセキュリティ設計パターンでは実装におけるコードレベルでの具体的な脆弱性への対策は対象外であるという問題がある。

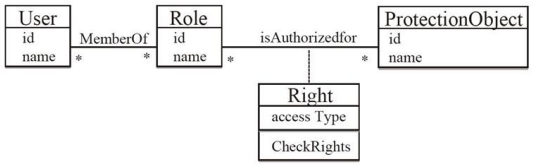

図 1　ロールベースアクセス制御パターン

3. 提案手法

3.1. 提案手法の全体像

提案手法の全体像を図 2 に示す。提案手法では、設計時に使われたセキュリティ設計パターンからテストテンプレートを作成し、テンプレートに設計の情報(振る舞いやクラス名、メソッド名など)を与えることでセキュリティ設計パターンの適用の妥当性検証のテストの生成を行う。テストテンプレートに Aspect 指向言語を用いることによって実装時の内部処理の観測を行い、テストによる検証を可能にする。実装に対して生成されたテストを行い、修正を繰り返すことによりセキュリティ設計パターンの適用検証を行う。

具体的な手順を以下に示す。

手順 0 テストテンプレートの作成
　テストテンプレートは事前にセキュリティ設計パターンから作成する。

手順 1 テストテンプレートの具体化
　テストテンプレートに設計の情報を与えることによってテストを作成することができる。

手順 2 設計の実装
　セキュリティ設計パターンが用いられた設計の実装

†1,2,3 早稲田大学

を行う。しかし、実装においてセキュリティ設計パターンが正しく適用できているのかここでは確認できない。

手順 3 パターンの適用検証

TDD(Test Driven Development)に基づき、作成されたテストを用いて実装のセキュリティ設計パターンの適用検証を行う。

手順 4 リファクタリング

手順 3 で発生したエラーに対してリファクタリングを行う。

手順 5 パターンの再適用検証

リファクタリングされた実装を再度セキュリティ設計パターンの適用検証を行う。もしテストが成功した場合、セキュリティ設計パターンは実装に適切に適用できていることが確認できる。そうでなければ再度テストに成功するまで手順 4 を繰り返す。

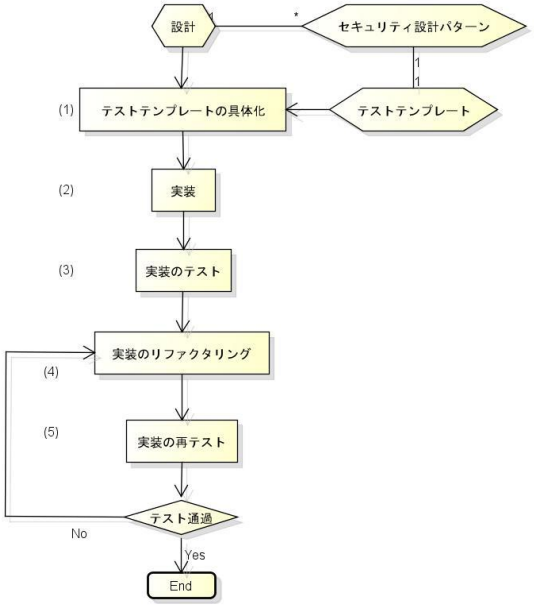

図 2 提案手法の全体像

3.2. アスペクト指向言語

アスペクト指向とは、各プログラム(モジュール)から共通に利用される機能のことであり、この機能はさまざまなモジュールにおいて横断的に利用される。そして、モジュールの機能とは独立している [4]。

本研究では、AspectJ というアスペクト指向言語の Java 実装を用いてテストテンプレートを作成する。AspectJ ではモジュールをクラスとしてではなく、アスペクトとして定義し、アスペクトの要素としてポイントカットとアドバイスを持つ。

本研究では、AspectJ によって実装時の内部処理の観測を行い、テストによる検証に用いる。

3.3. テストテンプレート

テストテンプレート作成の手順は以下通りである。

1. セキュリティ設計パターンの定義よりディシジョンテーブルの作成。
2. ディシジョンテーブルとセキュリティ設計パターンで定義された構造より実装時の内部処理を観測するためのアスペクトのテンプレート作成。
3. ディシジョンテーブルとセキュリティ設計パターンで定義された振る舞いよりテストケースのテンプレート作成

テンプレートには「アスペクトのテンプレート」と「テストケースのテンプレート」があり、これらのテンプレートに設計の情報を与え、テストを作成する。ディシジョンテーブルとは複数の条件の組み合わせに対しての動作を一覧にしてまとめた表のことである。

4. おわりに

本稿では AspectJ を用いたテストによるセキュリティ設計パターンの適用確認を提案した。セキュリティ設計パターンは抽象的なパターンであるために実装にどう落としていけばわかりにくいが、本稿ではテストを先に与えテスト駆動開発を行うことで実装を支援し、パターンの適用確認を行う予定である。具体的には、テストテンプレートに設計情報を与えることでテストを生成し、そのテストを満たすように実装を行っていくことでセキュリティ設計パターンの適用確認を可能とした。

今後の課題としては、どのようなパターンにおいてこの手法が可能であるのか検証する必要がある。

参考文献

[1] N.Yoshioka, H.Washizaki and K.Maruyama, “A Survey on Security Patterns” Progress in Informatics, No.5, pp. 35-47, 2008.

[2] M.Schumacher,E.Fernandez-Buglioni, D.Hybertson, F.Buschmann and P.Sommerlad, “Security Patterns”, Wiley, 2006.

[3] 吉岡信和, “セキュリティの知識を共有するセキュリティパターン”, 情報処理 52(9), 1134-1139, 2011.

[4] 長瀬嘉秀、天野まさひろ、鷲崎弘宜、立堀道昭, “AspectJ によるアスペクト指向プログラミング入門”, ソフトバンククリエイティブ,2004.