6W-06

ファジングツールの機能評価と診断パターンファイルの特性調査

福山潤†　　松井俊浩†

情報セキュリティ大学院大学　情報セキュリティ研究科†

1. はじめに

ファジングは、検査対象に対してファズと呼ばれる不正なパケットを大量に送信し、その応答や挙動を監視することで脆弱性を検出する脆弱性検査であるが、ファジングツールの種別は多く、ユーザはどのファジングツールを使って、どの部分を検査すればよいかが不明瞭であるのが現状である。またファジングを網羅的に実施するためには適切な診断パターンファイルの記述が必要不可欠であり、同ファイルの記述方法の困難さが、ファジングツール使用の困難さへ直接結びついていると言っても過言ではない。本稿では、既存のネットワークファジングツール及び最新のファジングツールについての機能評価を行いながら、診断パターンファイルの記述方法について調査した。

2. 関連研究

以前から存在し、現在も頻繁に更新が続けられているネットワークファジングツールには、Peach や Sulley といったツールが存在するが、M.Sutton ら (2007)[1] は、GeneralPurposeFuzzer, DFUZ, AUTODAFE, SPIKE 等の複数のファジングツールについて、動作対象 OS や、開発言語、ファジングを行う際のコードの再利用可能の有無や、設定変更のし易さ、ネットワークモニタ機能の有無等、複数の視点からその機能評価を行っており、その中でも Sulley は SPIKE の後継として、ターゲットの挙動監視や復旧機能を備えた、非常に優秀なファジングツールであると評価している。また Peach の製作者である Michael Eddington (2009)[2] は、Codenomicon や beSTORM といった商用ツールも比較対象に加え、診断パターンファイルの検証機能や、クラッシュ分析機能、ファジングツール自体の存続期間等、ツールの信頼性についても比較対象項目とし、多くのツールの機能比較を行いながら、ファジングによってダウンしたターゲットを復旧する機能を持つ peah や sulley に焦点をあて評価している。

Function evaluation of fuzzing tool and characteristics investigation of diagnostic pattern file
†Jun Fukuyama and†Toshihiro Matsui
†Institute of Information Security

3. 最新のファジングツール

昨今では、Internet of Things (IoT)技術の発展により、ネットワーク接続機能を持つ数多くのデバイスが次々と開発、販売されており、ファジングによる脆弱性検査の必要性は、以前にも増して高まってきている。その中でも boofuzz[3]は sulley の後継モデルをうたって開発されているファジングツールであり、python で動作する GPL2 ライセンスのツールである。sulley のバージョンアップが滞っているのに対し、boofuzz はかなりの頻度で更新が加えられている。標準的な特徴は sulley のそれらを踏襲しつつも、オンラインドキュメントの充実やインストールの容易性、CSV 出力機能を備え、またバグが少ないことがメリットとして挙げられている。

4. 機能比較

評価手法としては、関連研究を踏襲し、peach, sulley, boofuzz それぞれのツールが持つ機能及び更新頻度について評価を実施した。なお、peach については商用版と MIT ライセンスのものと分かれている為、MIT ライセンス版を比較対象とした。

4.1 機能評価

それぞれのツールは、標準的な機能は既に備わっており、機能面での優劣はつけがたかった。顕著な違いは診断パターンファイルの記述方法とライセンス種別であり、peach が XML/MIT ライセンスなのに対し、sulley/boofuzz は python/GPL ライセンスであった。また、sulley と boofuzz の診断パターンファイルについては、非常に似通ったファイルを用いるが、互換性は無い。（図１）

4.2 更新頻度評価

OSS はツール自体の透明度と、更新が容易であるのがメリットである。これら３つのツールについての公開日、及びバージョンアップ回数、最終バージョン公開日について調査を行ったところ、sulley は一度も更新がされていないのに対し、boofuzz は 7 回ものツール更新があった。この結果は直接ツールの有用性には直結しないが、ツール選択の一つの指標となり得る。（図２）

Copyright ©2018 Information Processing Society of Japan.
All Rights Reserved.

ツール	動作対象	パターンファイル	挙動監視	クラッシュ分析	オンラインドキュメント	ターゲットの自動再起動	その他	ライセンス	OSS
peach	win/linux	XML	✓	✓	✓	✓	診断パターンファイル検証	MIT	—
sulley	win/linux	python	✓	✓	✓	✓	—	GPL	✓
boofuzz	win/linux	python	✓	✓	✓	✓	CSV 出力	GPL	✓

図１ファジングツールの機能比較

ツール	リリース	更新頻度	最終更新
peach	2007 / 07	17 回	2014 / 11
sulley	2014 / 06	0 回	2014 / 06
boofuzz	2016 / 07	7 回	2017 / 11

図２　ツールの更新頻度(pre-release を除く)

boofuzz については、リリース間近という事もあり、サンプルパターンファイルが非常に少なく、利用者は一から作成しなければいけないのが現状ではあるが、更新頻度やコミュニティサイトが非常に充実しており、OSS であることも相まって、今後の発展が予想される。しかし現在の状況であれば、サンプルが豊富な peach や sulley の使用を検討すべきであろう。

5. 動作検証

前述の通り、sulley や peach はほぼ同等の機能を持っており、綿密なファジングができるか否かは、読み込む診断パターンファイルによって左右される。では、同様の内容のパターンファイルを読み込ませた場合、これらのツールに差異は生じるのだろうか。Protean Security[4]は、fuzzyftp と名付けた peach 及び sulley 用のパターンファイルを作成し、既知の脆弱性の存在する FTP サーバに対しファジングを行い、その結果、sulley では検知できなかったものが、peach で検知できたという結果となった。では peach で検出できたものを sulley で検出することは不可能なのだろうか。本章では、それぞれに作成した診断パターンファイルの記述について比較を行った。

5.1 対象のサービスと診断パターンファイル

検証にあたっては、Protean Security の研究結果にある既知の脆弱性 CVE-2006-2961 を持つ FTP サーバソフトウェアである Cesar FTP 0.99g をファジング対象として、診断パターンファイル fuzzyftp を使用し peach と sulley 双方の結果の比較とそれぞれに用いたパターンファイルの比較を実施した。

5.2 診断パターンファイル検証結果

本件に使用した sulley のパターンファイルで、文字列のファズデータを作成する s_string(‘fuzz’) プリミティブは、デフォルト設定のまま使用すると、ascii コードで表現できる文字列のみを生成し、例えば UTF-8 で表せられる特殊文字等でのファズデータは作成されないことが判明した。このように、ファジングを行う際には、診断パターンファイルの記述に左右されるところが多く、パターンファイルの記述方法は勿論、ファジング対象のプロトコルについても深く学習する必要がある。

5.3 診断パターンファイル設定言語の比較

peach と sulley の診断パターンファイルについて比較すると、peach の XML での記述は、それぞれの項目について細かく記載する必要があるため、非常に大量の記述が必要になる。それに対し、sulley の python での記述は、ファイルごとに細かく区切られている為、多くのコードの流用が可能であり、使用者は少ない記述でファジングを実行できるのが利点ではあるが、設定項目が分散して存在するため、その構造については深く理解する必要がある。

6. まとめと今後の課題

本稿では、既存のファジングツールと新たなファジングツールについての機能面での比較と、既存のファジングツール 2 種における動作結果から判明した診断パターンファイルの比較を実施し、ファジングに用いるパターンファイルの重要性について調査を行った。

今後の課題としては、peach や defensics の商用ツールとの機能比較や、ツール間をまたぐ診断パターンファイルの変換方法について研究を行い、既存のパターンファイルを新しいファジングツールのパターンファイルへ変換する方法等、使いやすさの向上に焦点をあて研究を実施していきたい。

参考文献

[1] M.Sutton, A.Greene, P.Amini, *Fuzzing: Brute Force Vulnerability Discovery* Addison-Wealey pp351-417 ,2007
[2] M.Eddington *Demystifying Fuzzers*, blackhat USA, 2009
[3] J. Pereyda boofuzz *Documentation Release 0.0.7* https://github.com/jtpereyda/boofuzz, 2017
[4] Protean Security fuzzyftp , https://github.com/proteansec/fuzzyftp

Copyright ©2018 Information Processing Society of Japan.
All Rights Reserved.