

ソースコードファジングツール Mayhem

ソースコードの潜在的なバグや脆弱性を検出し、ソフトウェアのセキュリティや堅牢性を向上させます

① ソースコードの動的解析（ファジング）を実行

- Mayhem プラットフォーム上でファズの生成、アプリケーションの実行、ファズの送信を繰り返すことで、動的にアプリケーションを解析します。
- 解析結果を GUI 上でリアルタイムに確認できます。
- あらゆるテストケースを再現してテストするため、偽陽性はありません。

② シンボリック実行技術※をファズ生成に応用

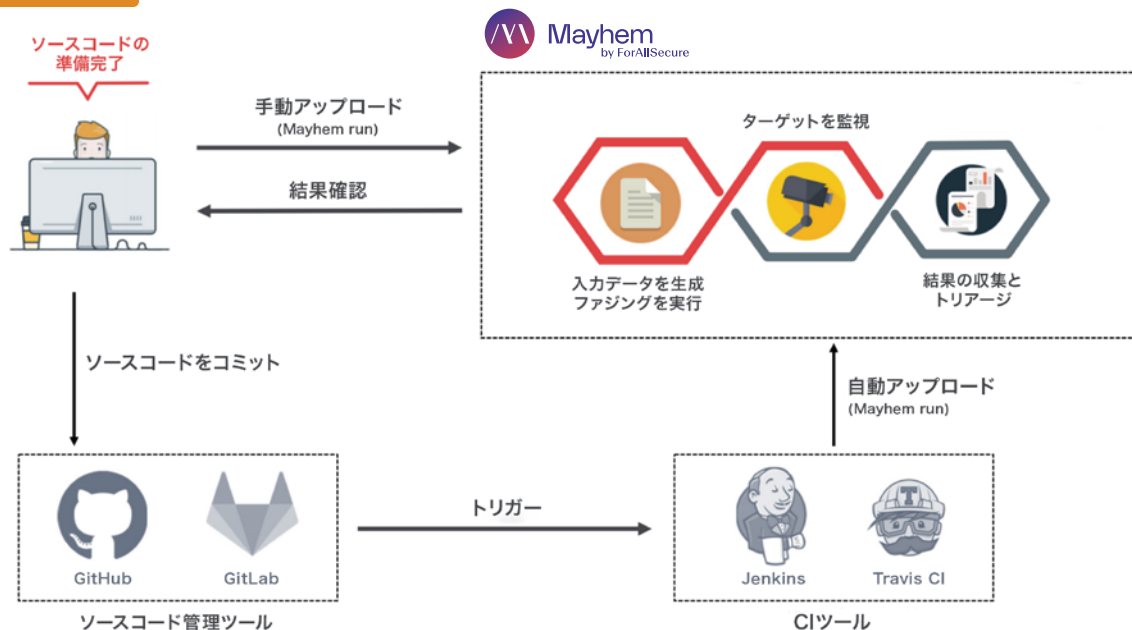
- より深いパスまで解析することができます。

※シンボリック実行(Symbolic execution)によるファズ生成エンジンは、カーネギーメロン大学の研究による特許技術を元に開発されたものです。

③ 開発の初期段階からソフトウェアセキュリティの実装が可能

- 動的テストのシフトレフトにより、開発の初期段階からバグや脆弱性を発見することで修正コストを削減できます。
- DevSecOps を念頭に置いて開発されており、さまざまな CI/CD ツールと連携できます。

ご利用イメージ



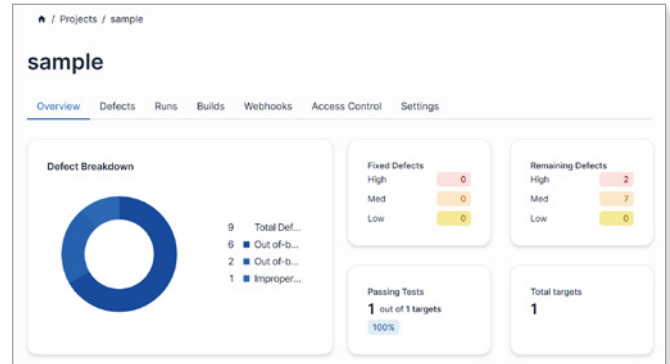
APIファジングを実行するオプション機能もございます。

主な機能

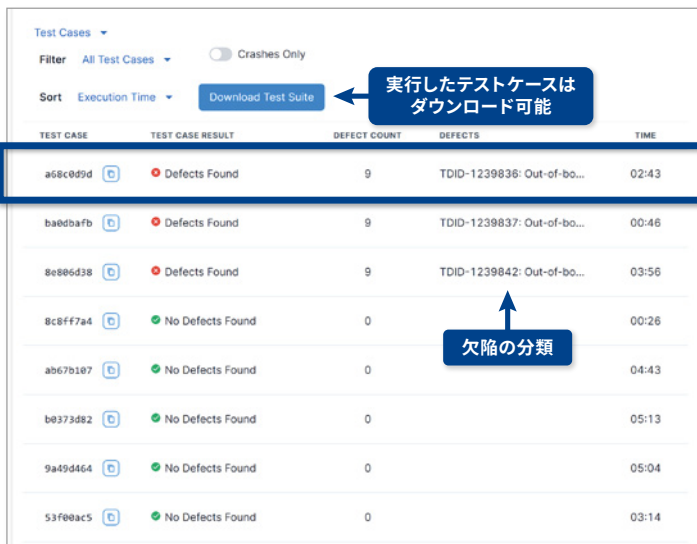
ファジングの進行状況と検出された欠陥をリアルタイムで確認できます。



見つかった欠陥情報のサマリーをダッシュボードで確認できます。



実行したテストケースを一覧で確認することができます。



実行したテストケースはダウンロード可能

TEST CASE	TEST CASE RESULT	DEFECT COUNT	DEFECTS	TIME
a68c0d9d	Defects Found	9	TDID-1239836: Out-of-bo...	02:43
baedba7b	Defects Found	9	TDID-1239837: Out-of-bo...	00:46
8e986d38	Defects Found	9	TDID-1239842: Out-of-bo...	03:56
8c9ff7a4	No Defects Found	0		00:26
ab67b107	No Defects Found	0		04:43
b037d82	No Defects Found	0		05:13
9a49d464	No Defects Found	0		05:04
53f0eac5	No Defects Found	0		03:14

欠陥の分類

各テストケースの詳細を確認することができます。欠陥を再現するコマンドが提供されるため、再現確認を容易に実施することができます。また、テストケースの実行時の標準出力やパケットレース等の情報が提供され、欠陥の原因の特定に役立てることができます。



Test Case a68c0d9d

Steps to Reproduce

- Download the test case file:
- Place the file in your working directory.
- In your terminal, run the following command:

```
./usr/local/sbin/lighttpd -D -f /usr/local/etc/lighttpd.conf & sleep 2; nc 127.0.0.1 80 < a68c
```

欠陥を再現するコマンド

Test Case a68c0d9d

Defects (9) | Triage | Advanced Triage (10)

Test Case: 入力されたファズデータ

```
000000: 4745 5420 2f20 4854 5450 2f31 2e31 0d0a GET / HTTP/1.1..
000010: 5573 6572 2041 6765 6e74 3a20 6375 726c User-Agent: curl
000020: 2f37 2e33 352e 300d 0a55 7365 722d 4167 /.7.35.0..User-Ag
000030: 656e 743a 2063 7572 6c2f 372e 3335 2e30 ent: curl/7.35.0
```

標準出力と標準エラー出力

```
Standard Out: (empty)
Standard Error:
2023-07-29 07:02:11: (log.c.75) server started
```

実行時に呼び出された関数のログ

```
#0 0x7ffff7e3d7bb in __GI_raise at /build/glibc-fWwX8/glibc-2.28/signal/../sysdeps/unix/s
#1 0x7ffff7e28535 in __GI_abort at /build/glibc-fWwX8/glibc-2.28/stdlib/abort.c:81:7
```

ファジングのカバレッジ情報が提供されます。他ツールと組み合わせることでカバレッジをさらに可視化することもできます。



ソースコードファジングサービスの実施手順 (Mayhemを使用)

お見積り	キックオフ	ファジング計画	ファジングの実行	結果の確認 & レポート
<ul style="list-style-type: none"> 要件をヒアリングの上、お見積りいたします。 ※ツールの選定なども含みます。 	<ul style="list-style-type: none"> スコープを確定します。 テストの実施方法や、監視方法、終了要件を設定します。 	<ul style="list-style-type: none"> 決定したスコープや要件に基づき、テスト計画を作成いたします。 要件を網羅するのに十分なファズデータを準備します。 	<ul style="list-style-type: none"> ファジングを実行します。 実行結果やパフォーマンスに応じてチューニングを行います。 	<ul style="list-style-type: none"> テスト結果の確認を行い、脆弱性が検出された箇所をレポートします。 再利用可能なテストケースは次回の実行に活用できるようDB化します。

テスト対象はLinux上で実行可能なバイナリファイルに制限させていただいています。

APIファジングサービスも同様の手順でご提供できます。

