HNG: Fuzzer Ideas

Table of Contents

[Main goal 2](#_Toc197936677)

[Product Ideas 2](#_Toc197936678)

[All in One Fuzzer (Original HNG Fuzzer PRD) 2](#_Toc197936679)

[Fuzzer Framework with 3rd party tools 2](#_Toc197936680)

[Glossary 4](#_Toc197936681)

# Main goal

Provide fuzzing platform which offers the following main features:

1. N-tier control
2. Crash detection
3. Crash-dump generation
4. [Defect co-relation](#_Glossary)
5. Report generation
6. Visual Studio \ TFS Integration

# Product Ideas

We currently have the following ideas for the fuzzer:

### All in One Fuzzer (Original HNG Fuzzer PRD)

The original design calls for a product which meets all the main goals described above and also provides:

1. Extensive library of built in fuzzers
2. Plugin model to create custom fuzzer
3. Protocol recognition
4. File format recognition
5. Template model for extending protocol and file format recognition
6. Online Community for sharing custom fuzzers and format\protocol templates.
7. [Black-Box API fuzzing](#_Glossary)
8. [Asset Virtualization](#_Glossary)

This tool will be a huge undertaking as we will have to develop technology for fuzzing techniques and template parsing. We will also have to create and ship a library of templates. Another disadvantage is that we will be putting yet another fuzzing tool out in the market which people will have to learn how to use.

### Fuzzer Framework with 3rd party tools

This product also meets all the main goals described above but unlike the original Fuzzer idea, does not provide its own fuzzing mechanisms or format\protocol recognition; instead this tool relies upon numerous 3rd Party fuzzing tools in the industry to actually fuzz the. The following features are proposed:

1. Extensive library of links to 3rd party Fuzzers
2. Plugin model to link 3rd Party Fuzzers to our Framework
3. Online Community for sharing HNG plugins for 3rd Party fuzzers, custom fuzzers and format\protocol templates for 3rd Party fuzzers
4. Firefox like Add-on manager for downloading and managing 3rd Party plugins and fuzzers
5. Recommendation engine for suggesting which 3rd Party fuzzers to use for a given scenario
6. [Buffered Fuzzing](#_Glossary)
7. [Multi Interface Fuzzing](#_Glossary)
8. [Asset Virtualization](#_Glossary)

This tool essentially provides an advanced framework for executing fuzz tests, detecting crashes and generating reports.

The main advantages that our framework will provide compared to directly using the 3rd Party fuzzers are:

1. **3rd Party tool discovery**: Not everyone knows what tools are out there. Our Plugin Manager will allow people to discover what fuzzing tools exist in the various fuzzing categories. We could also rank tools for specific uses based on a **Community Rating** mechanism.  
   Example User Scenario: User knows about the assets which require fuzz testing but is not sure which fuzzer to use and what fuzz options to select. Using HNG’s Fuzzer database and Community Ratings we suggest Tools and options to the user.
2. **Multi Interface Fuzzing**: Most small 3rd Party tools were developed to interact with limited interfaces (usually only one). Since we have control of the application, we can take the output from the 3rd party tool and apply it to many interfaces in the Test Application. An example of this is [Black Box API Testing](#_Glossary).  
     
   Example User Scenario: The user wants to use a specific 3rd party fuzzer because of its advanced fuzzing technology. However, that fuzzer only fuzzes files and the user wants to fuzz Pipe data. HNG Fuzzer can redirect the fuzzed output from the 3rd party tool to the pipe.
3. **Buffered Fuzzing**: 3rd Party tools may be good at understanding protocols and have various fuzzing options but they are limited by how they fuzz. I do not think they have the ability to fuzz entire data sets and instead fuzz data chunks as they come in.  
     
   Example User Scenario: A user wants to use a 3rd Party fuzzing tool to fuzz an XML file that is being received over the network by their Test Application. However, the 3rd party network fuzzer can only see the XML in chunks as it streams over the network – this limits the ability of the fuzzer to get a bigger picture of what is being fuzzed. The HNG Fuzzer will buffer the received data until it has received the entire XML, then fuzz it with the 3rd Party tool and stream the result to the Test Application.
4. **Asset Virtualization**: We do this in Holodeck Enterprise Edition already and currently don’t know of any other fuzzing tool in the market which can perform real-time fuzzing without affecting the original file itself. We have an advantage due to our interception technology. **Example User Scenario**: The user wants to fuzz a file which is used by more than one application. However, the user only wants to test the effects of the fuzzed file on a specific Test Application and not the others. HNG Fuzzer will allow the user to fuzz the data from the file and feed it to the Test Application without affecting the other applications.
5. **All features mentioned in** [Main Goals](#_Main_goal).

# Glossary

Asset Virtualization: The ability to fuzz a resource in isolation without physically altering it so that other applications which depend on that resource are not affected by it.

Black Box API Fuzzing: The ability to fuzz API interfaces in a black-box environment. Since we have the ability to intercept APIs we can use the output of 3rd Party tools and feed the fuzzed data as parameters to APIs.

Buffered Fuzzing: The ability to fuzz a data stream (synchronous or asynchronous) as a whole and not in parts as the data arrives. <Assumption>This is a huge weakness in the current fuzzing tools which fuzz network data. They fail to see the data as an entire set and fuzz only chunks of it as it becomes available.<\Assumption>

Defect co-relation: The ability to co-relate application crashes, exceptions and errors with test events (fuzzing in this product) and system event logs in order to provide a larger picture of the effects of testing.

Multi Interface Fuzzing: The ability to take the fuzzed output from a 3rd Party fuzzer and apply it to more interfaces than what that 3rd Party tool natively supports.