Learning the Peach Fuzzer

Dr. Jared DeMott
CTO AND FOUNDER

@jareddemott www.vdalabs.com

Overview

Peach

- Data model
- Operation
- Verifier

010 hex editor

Demo

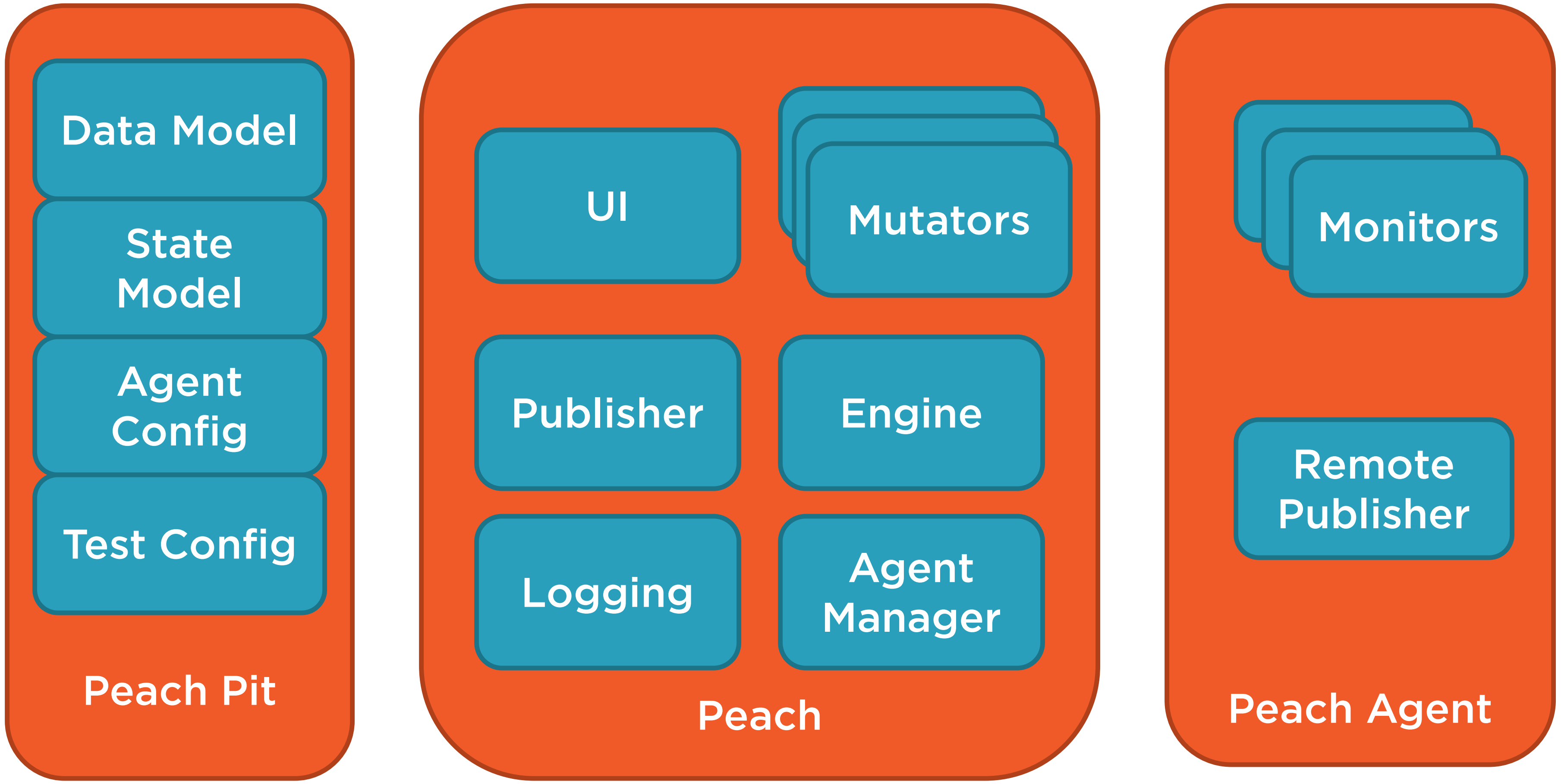
Fuzzing with Peach community

Peach Setup

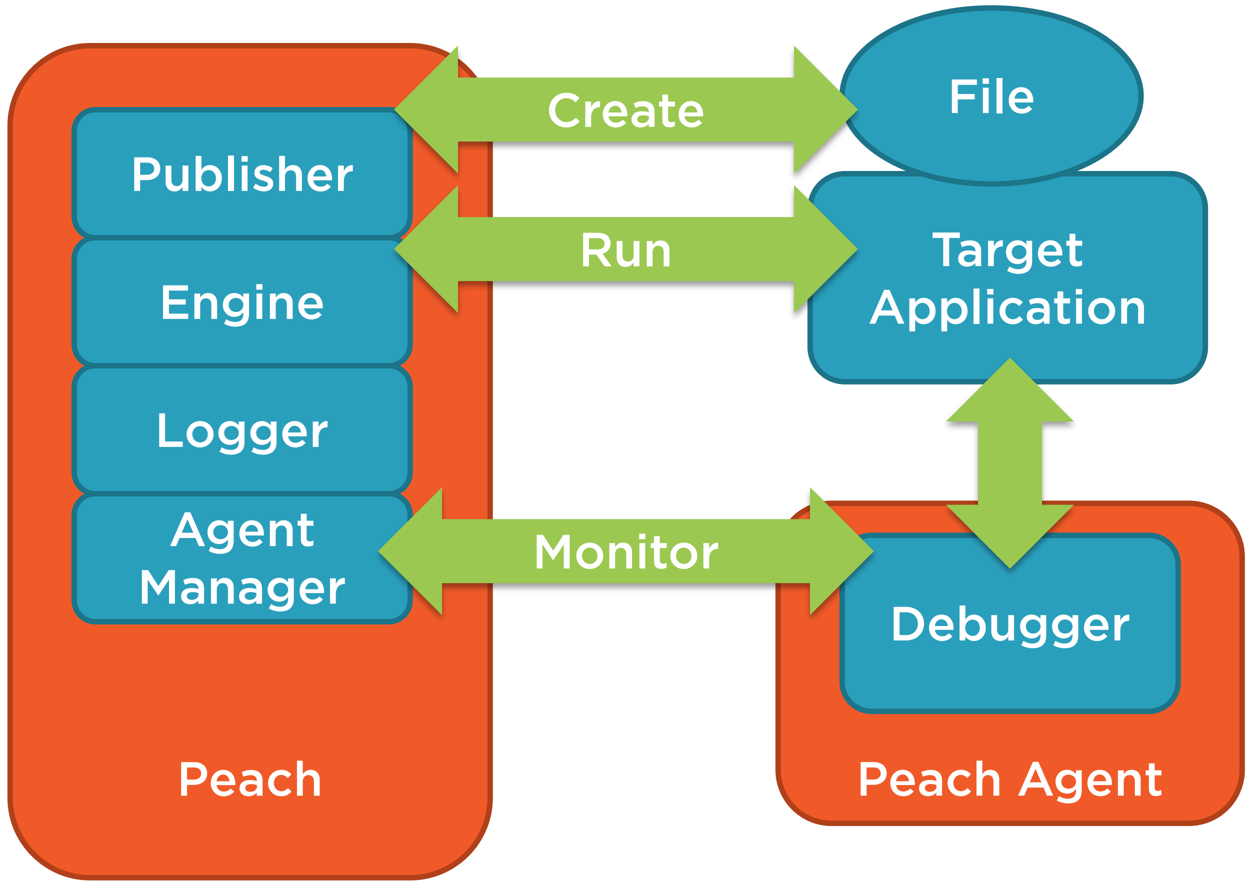
http://www.peachfuzzer.com/resources/peachcommunity/

- Windows
  - Install Microsoft.NET v4 Runtime
  - Install Debugging Tools for Windows
  - Unzip Peach binary distribution to a working folder
  - Your now ready to start using Peach 3!

Peach Components



File Fuzzing Diagram for Peach



Building a Peach Pit

XML editing

- Visual Studio is a good editor
  - oXygen and XML spy are others

Common elements

- Param
  - Parameter to a parent element
- Value
  - Literal string like “\r\n”

Peach Documentation

- Online or in docs folder

```
<?xml version="1.0" encoding="utf-8"?>

<Peach xmlns="http://phed.org/2008/Peach"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

    xsi:schemaLocation="http://phed.org/2008/Peach  ../peach.xsd"

    version="1.0"

    author="Michael Eddington"

    description="DHCP Request Fuzzer">


    <Include ns="default" src="file:defaults.xml" />


</Peach>
```

Data Model

Used to describe the data you wish to fuzz

- This is a very important step in intelligent fuzzing
- More than one can be defined

Composed of:

- Block
- Sequence
- String
- Number
- Flags/Flag
- Blob
- Relation
- Transformer

```
<DataModel name="UdpPacket">
    <Number name="SrcPort" size="16" endian="network" />
    <Number name="DestPort" size="16" endian="network" />
    <Number name="Length" size="16" endian="network">
        <Relation type="size" of="Data" />
    </Number>
    <Number name="CheckSum" size="16" endian="network>
        <Relation type="checksum" of="UdpPacket" />
    </Number>
    <Blob name="Data" valueType="hex" value="41 42 43 44"/>
</DataModel>
```

Common Attributes

Name

- Element name

ValueType and value

- Assign a default value

MinOccurances and maxOccurances

- Occurrences of an element

Ref

- References an already defined element

State Model

Describe the protocol by which data will be sent to/and from target

State Models are comprised of *States*

- Must be provided an initial state

States are comprised of one or more actions

- Each action has a type
  - Input, output, etc.
  - Steps include actions like writeFile, close file, etc

```
<StateModel name="State" initialState="Initial">
        <State name="Initial">
            <Action type="start" />
            <Action type="connect" />

            <Action type="output">
                <DataModel ref="DhcpRequest" />
            </Action>

            <Action type="close" />
            <Action type="stop" />
        </State>
    </StateModel>
    <Agent name="LocalAgent" location="http://127.0.0.1:9000">

        <Monitor name="Debugger" class="debugger.WindowsDebugger">
            <Param name="Command"
value="C:\Peach\samples\CrashableServer\CrashableServer.exe"/>
            <Param name="Params" value="192.168.1.195"/>
        </Monitor>

        <Monitor name="Network" class="network.PcapMonitor">
            <Param name="filter" value="tcp"/>
        </Monitor>
    </Agent>
```

Publishers

Provide I/O interfaces

Two basic types

- Stream based
  - TCP, UDP, FILE
- Call based
  - COM, Shared Library, RPC

Remote Publishers via Agent System

Can add new publishers

- Fuzz an embedded system?

```
<Test name="DhcpRequestTest">

      <StateModel ref="State" />

      <Publisher class="udp.Udp">

            <Param name="Host"
  value="192.168.1.10" />

            <Param name="Port" value="67" />

      </Publisher>

  </Test>
```

```
<Run name="DefaultRun">
    <Test ref="DhcpRequestTest"/>

  <Logger class="logger.Filesystem">
        <Param name="path" value="c:\peach\logtest" />
    </Logger>

</Run>
```

Faults

Log folder

- Contains information gathered from the monitors as well as the file that caused the fault

Summary

Fuzzing a target with Peach

- Helper utilities

How to scale