FORTRA

beSTORM
13.2.0
Quick Start Guide

202403110931

Table of Contents

Introduction

beSTORM represents a new approach to security auditing. It's essentially a fuzzing framework that can be used for securing in-house developed applications and devices, as well as applications and devices of external vendors. Vendors can use beSTORM to test their products in a certification test or as part of the development life cycle. It provides the ability to customize existing modules and add new modules for testing all in an intuitive and easy to use environment.

Although beSTORM is a generic fuzzing framework, no programming skills are necessary to use it. It is especially useful for testing standard protocols – HTTP, POP3, SMTP, SIP and similar protocols with an RFC definition as well as standard file types – BMP, TGA and similar. Of course, you can always define your own testing modules to test proprietary protocols.

> **NOTE:** Although most of the examples in this document will refer to network protocols, beSTORM is certainly not limited to testing network protocols alone. Network protocols can be seen as “recipes” to building anything from an HTTP protocol description traffic to a JPEG file description

System Requirements

The following are the hardware and software system requirements for beSTORM:

Hardware

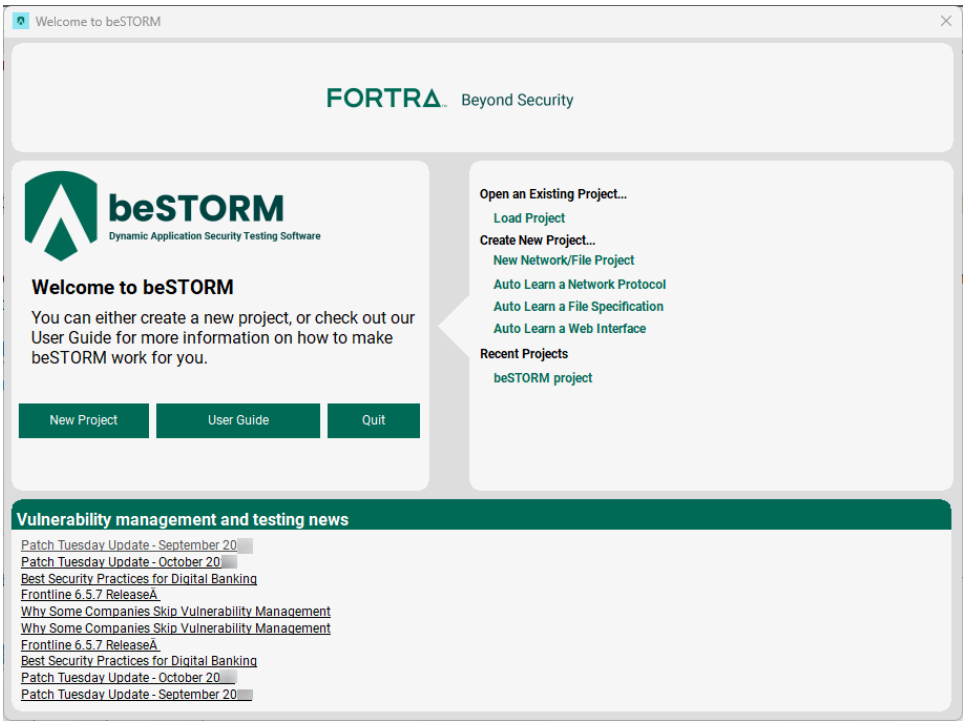
Hardware component	Minimum requirements	Recommended requirements
Processor	x86-64 processor	Quad-core processor (Intel i5+ or equivalent)
Memory	1 GB RAM (Linux, Docker, Embedded application)	8 GB RAM (Windows 10)
Hard drive	250 MB available hard drive space	1 GB available hard drive space

Software

Operating system	beSTORM version(s)
Windows 10 or later	13.1.0 or later
Kali (rolling)	11.6.27, 10.9.16, 10.7.23
Ubuntu 20.04	11.5.23
Ubuntu 18.04	10.10.19

Welcome to beSTORM Screen

When you first launch beSTORM, you are presented with a Welcome to beSTORM screen, from which you can begin working with beSTORM and utilize its different features.



The welcome screen allows you to either open up an existing project, by way of Load Project, create new projects, by way of New Network/File Project, teach beSTORM about new standards, by way of Auto Learn a Network Protocol or by way of Auto Learn a File Specification and finally use beSTORM to test your product's API, by way of Auto Learn a Web Interface.

In addition, the welcome screen provides shortcuts to previously loaded beSTORM projects.

Getting Started with beSTORM

beSTORM utilizes a wizard interface that guides you through the process of configuring beSTORM's testing session, fuzzing optimizations, and monitoring capabilities.

To begin your fuzzing session:

1. Open **beSTORM Client**.
2. Select **New Project**.
3. On the **Welcome** page, configure the following:
    a. **Project Name** - Enter a name for the project, or use the default name provided.
    b. **Location** - Browse to a location to store the project and its files, or use the default location provided.
    c. **Wizard level** - Select **Simple** to use pre-configured settings, or select **Advanced** to manually configure those settings yourself.
    d. **Perform a port scan, and service detection and assist me in choosing the relevant module** - Select or clear this setting, based on your preference.
4. Select **Next**.
5. On the **Basic Configuration** page, select a module for your project:
    a. **beSTORM's predefined modules** - Predefined modules such as; BMP, GIF, FTP, HTTP, POP3, etc.
    b. **Import a Custom Module from a BSM F**ile - A custom module you can import.
    c. **Build a Network Module** - Create a new Network-based module utilizing beSTORM's network auto learning capabilities.
    d. **Build a File Module** - Create a new File-based module utilizing beSTORM's file auto learning capabilities.
    e. **Build a Web Application Module** - Create a new Web Application module utilizing beSTORM's web testing capabilities.
    f. **Build a CANBUS Module** - Create a new Controller Area Network (CAN bus) module utilizing beSTORM’s ability to read and process CAN DBC files.

    Depending on your selection, the **Hostname or IP address** (default is **127.0.0.1**), **Protocol** (default is **udp**), and **Local Port** (default is **67**) parameters are preset. If you select a predefined module (which do not require network configuration) or a new File, Web Application, or CANBUS module, the Target Host Settings section of the wizard will not appear.
6. Select **Next**.

7. If you selected **Advanced** in step 3c, the **Advanced Configuration** page will appear providing optimizations and options based on the module selected in step 5 (*if you selected **Simple** in step 3c skip to step 11*). The available options are:
    a. **Optimizations** - Different modules support different settings; for example HTTP testing do not run multiple Parallel Attack Threads, while modules such as SMTP do. This depends on the type of server being tested, the robustness of the protocol to parallel testing, and other considerations.
    b. **Run in batch mode** - If selected, beSTORM continues running after the first fault is found.

    > **NOTE:** In this case, the product being tested should recover from the previous fault, either by being restarted or by some other way.

    c. **Make sure monitor is up before starting** - If selected, beSTORM waits for an agreed signal from the tested environment before beginning the test. This allows beSTORM to be certain that the tested environment is running properly.
    d. **Report connectivity issues as exceptions** - If selected, beSTORM treats a case of receiving no network traffic from the tested environment as potential problems or vulnerabilities. This feature is useful while testing an environment that is not easily monitored (such as a proprietary hardware device) as it marks all network problems as a potential vulnerability. By doing so, it allows to easily reproduce the issue at a later time and discover the cause.
    e. **Periodically test connection and report vulnerability upon failure** - If selected, this option tests the behavior of the product being tested and expects it to answer traffic that is not malformed in a normal manner. If the product does not respond, beSTORM reports an exception.

    Certain modules require additional information to performing proper testing. One such example is in the case of the FTP module, a username and password are needed for FTP login, if you want to test all the available commands.

    > **NOTE:** beSTORM can not log in without a proper username and password.

8. Select **Next**.
9. On the **Module Environment** page, some of the fields appearing here are automatically populated by values that were previously defined (for example, Remote Hostname, Remote Port, and Remote Protocol Type). Other items to note:
    a. To change the username or password value, double-click on the **Value** field just right of the Descriptive text Username for FTP login and Password for FTP login respectively.

b. The **Required** column indicates which parameters must contain an actual value. If the protocol contains such parameters, beSTORM prompts you to supply values, otherwise values are assigned by default.

10. Select **Next**.
11. On the **Extra Configuration** page, configure these test settings:
    a. To adjust the speed of your test to be a fixed number of sessions per seconds, select a value for **Saturation Rate Threshold** and leave **Fixed Saturation Rate Threshold** selected.
    b. To allow your test's speed to be automatically adjusted according to reports from the beSTORM monitor, select **Auto Adjust - Optimize CPU usage**.
    c. To configure the monitoring communication settings, optionally select from any of the following monitor options:
        i. **ARP Echo** – Attempts to resolve the IP address of the machine tested into a MAC address.

        > **NOTE:** ARP Echo properly works on LAN in a WAN environment where the target is not on the same network/subnet class. An ARP response will be received from the Router that connects the two networks, thus causing a false status.

        ii. **ICMP Echo** – Attempts to perform an ICMP Echo/ICMP Response test on the remote IP address.
        iii. **UDP Echo** – Attempts to verify whether the remote UDP port is open.

        > **NOTE:** For UDP to be properly detected as non-responsive/closed, the Windows Firewall has to allow ICMP Destination Unreachable packets to arrive.By default, Windows Firewall blocks such packets from arriving.

        iv. **TCP Echo** – Attempts to verify whether the remote TCP port is open.
        v. **External Monitor**- The Fortra's Beyond Security provided monitor, or your own custom monitoring device/program.

        Defined what is the IP address of the machine you would like to perform ARP, ICMP, UDP, or TCP monitoring, by providing a value to the Monitored IP address field, and if you are utilizing UDP or TCP, define what port you would like to monitor by providing a value to the Port field just right from the Monitored IP address field.
    d. Enter a port number for the **Incoming Command Port** parameter. This parameter is for receiving commands from the monitor (such as reports on the tested machine's load and status).

e. Enter a port number for the **Incoming Exception Port** parameter. This parameter is for Exception Data being sent from the monitor to beSTORM.

f. Enter a port number for the **Outgoing Command Port** parameter. This parameter tells beSTORM which port to use to establish that communication.

12. Select **Next**.
13. To prevent beSTORM from automatically scanning after completing the wizard, clear the **Auto-start beSTORM scan now** check box.
14. Select **Finish** to complete the wizard.

Fuzzing

beSTORM starts its fuzzing as soon as you select **Start**.

The fuzzing sequences are deterministic and can be replayed by telling beSTORM to start from the beginning, or from any other particular attack vector (position) you provide to beSTORM.

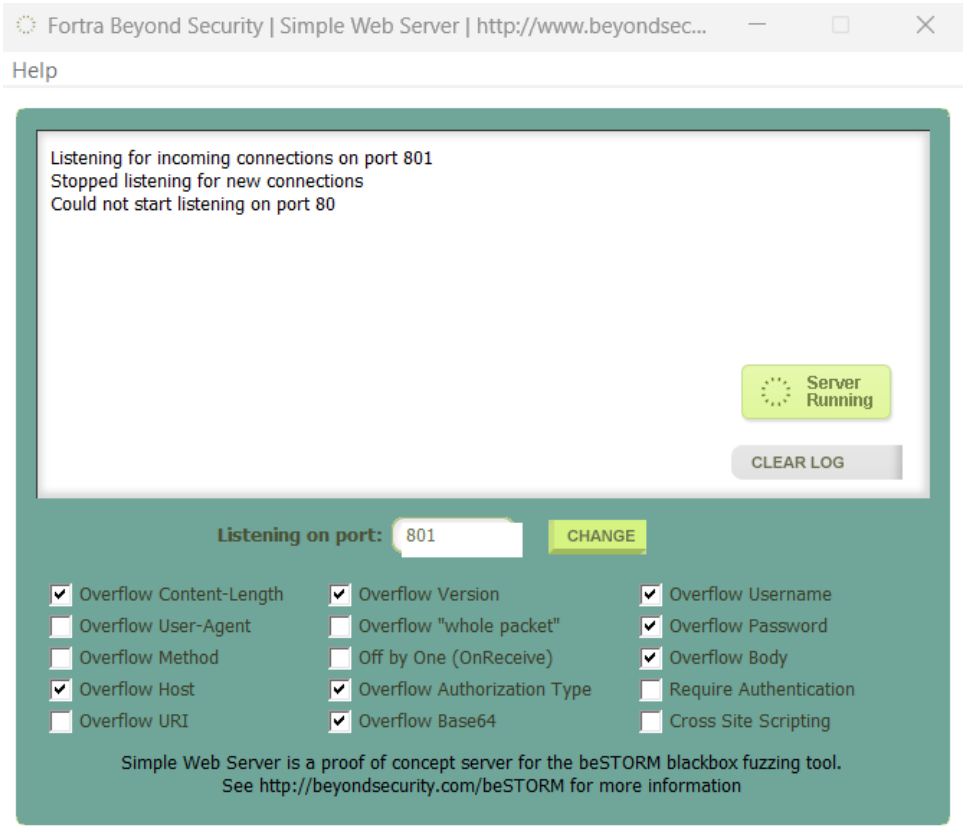
You can monitor beSTORM's progress by viewing the Progress Information section by selecting **Preview**. This displays the dataset currently being sent by beSTORM to the tested product, or you can look up the Detailed Log to view the current speed beSTORM is testing the product.

Even though beSTORM has predefined buffers which it fuzzes, you have complete control over the types of data it fuzzes and the type of data it generates (for example, long buffers, overflowing integers, etc.). Changing these predefined buffers, or even adding additional buffers, can greatly enhance the performance and the usability of beSTORM as it allows it to find more exceptions quickly, as well as find exceptions that might be specifically relevant to your product.

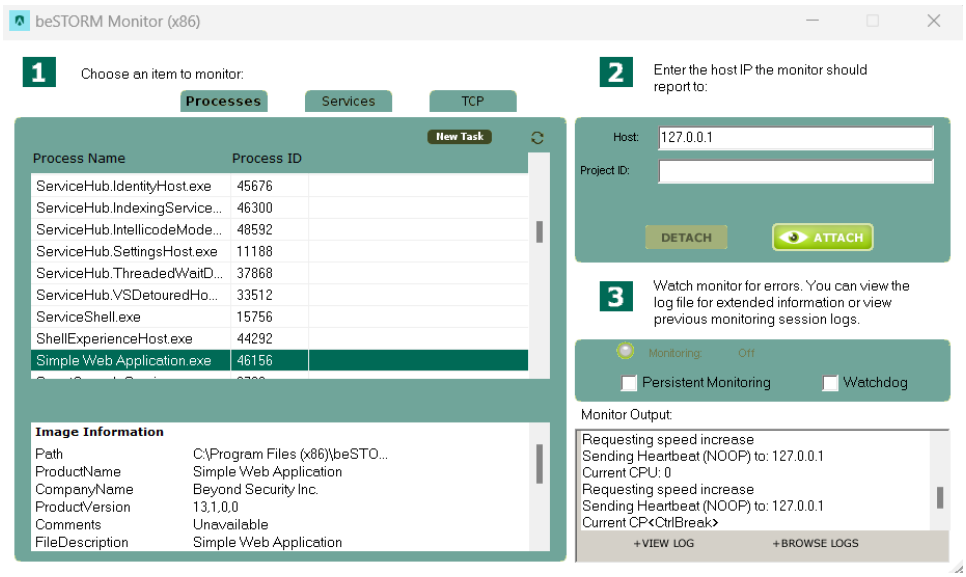
beSTORM Walkthrough

To demonstrate the initial stages of running beSTORM, the example below has a web (HTTP) server that has been tainted with numerous vulnerabilities such as: Overflow by way of Method, Overflow by way of URI, Overflow by way of User-Agent, Off-by-One in Receive, Overflow in Base64 decoded content, and others.

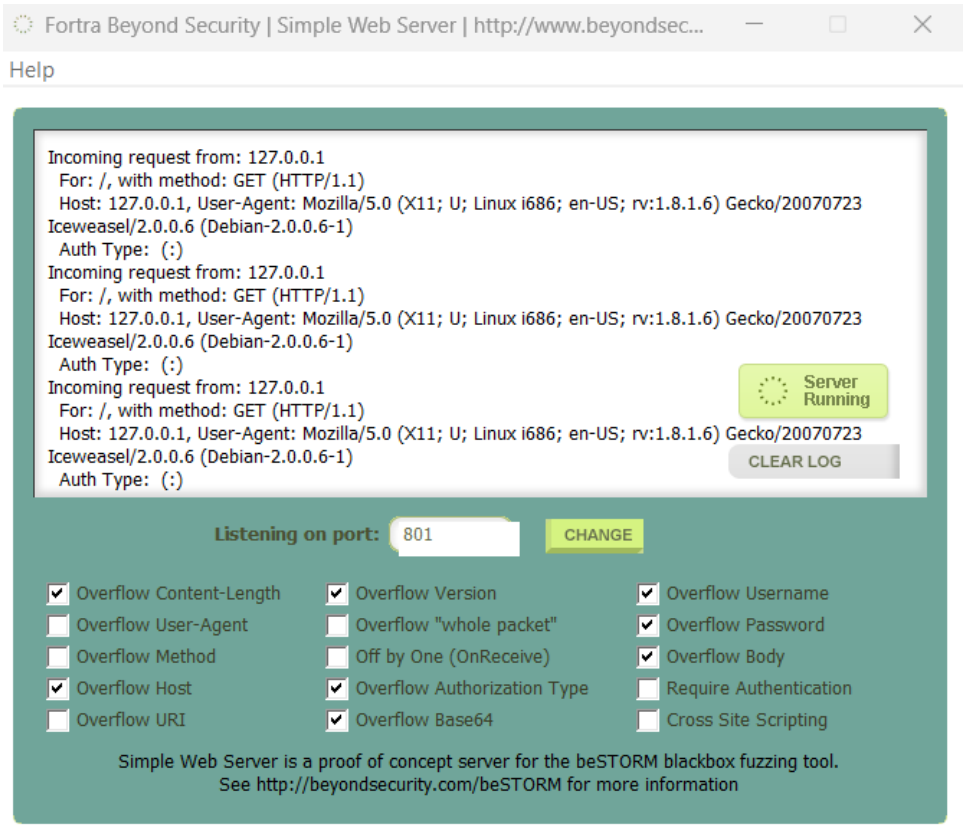
To use the web server, launch the executable and the program automatically starts to listen on the desired port.



To monitor the port and its status, use the GUI version of the monitor, shown below:



The Simple Web Server process (Simple Web Server.exe) has been selected from the process list on the left side, and a host to report to on the right top side (127.0.0.1). To begin monitoring, click **Attach**. The Simple Web Server automatically begins answering incoming requests:



You can quickly test whether the program crashes by sending the following request:

```
http://AABBCCDDEEFFAABBCCDDEEFFAABBCCDDEEFFAABB:AABBCCDDEEFFAABBCCD
DEEFFAABBCCDDEEFFAABB@ip_address/
```

If you inspect the traffic being sent using Google Chrome, you will see that this URL is sending a long Base64 authorization request:

The web server should crash, and if you inspect the log file of the monitor you should see something similar to the following:

```
[INFO] beSTORM Monitoring Agent version: 3.0.1

[INFO] setProgramName: C:\temp\Simple Web Server.exe

[INFO] CreateProcess was successful

[INFO] setProcessID: 1408

[INFO] OpenProcess successful

[INFO] OpenProcessToken successful

[INFO] LookupPrivilegeValue successful

[INFO] AdjustTokenPrivileges successful

[INFO] attachToProcess

[INFO] We created the process no need to re-debug it

[INFO] The process 1408 has been attached successfully

[INFO] EXCEPTION_BREAKPOINT

[INFO] Page fault on read access to 0x6846581d

[INFO] 00000580:000008f4: exception code=c0000005
```