

# Testing or Hacking: Real Advice on Effective Security Testing Strategies

Daniel Billing  
Medidata, UK

@thetestdoctor

 **EuroSTAR**  
Software Testing  
CONFERENCE

STOCKHOLM  
31.10 - 03.11  
2 0 1 6



# WE HAVE A PROBLEM

# OUR APPLICATIONS AND SYSTEMS ARE BEING HACKED





# TESTING CAN HELP

# WE NEED TO LEARN FROM THE HACKERS



**Keren Elazari**

**"Hackers...they  
might just be the  
immune system  
for the  
Information Age"**





# WHY DO WE NEED TO DO SECURITY TESTING?

# THAT'S OUT OF SCOPE

# WE ARE OUTSOURCING THAT

# THAT'S A NON FUNCTIONAL REQUIREMENT

# WE DON'T HAVE THE SKILLS



**WE NEED TO DELIVER FAST,  
NOT SLOW THINGS DOWN**

**I THINK WE NEED TO DO  
SOME SECURITY TESTING**

# WHERE DO WE START?



# HOW DO WE DO IT?



# WE NEED SOME TRAINING

# WHAT TOOLS DO WE NEED?

# HOW DO WE KNOW IF WE ARE SECURE?



# 1. CONSIDER THE SCOPE

# FUNCTIONAL FLOW

**REGISTER**

**SEARCH**

**RESULTS**

**PURCHASE**

**STOCK**

**ADMIN**



# UNREGISTERED USERS

**REGISTER**

**SEARCH**

**RESULTS**

# REGISTERED/ADMIN USERS

**PURCHASE**

**STOCK**

**ADMIN**





# 2. KNOW YOUR STACK





# ALL COMPONENTS HAVE POTENTIAL VULNERABILITIES



# POOR IMPLEMENTATIONS OF ANY COMPONENT CAN LEAD TO FLAWS



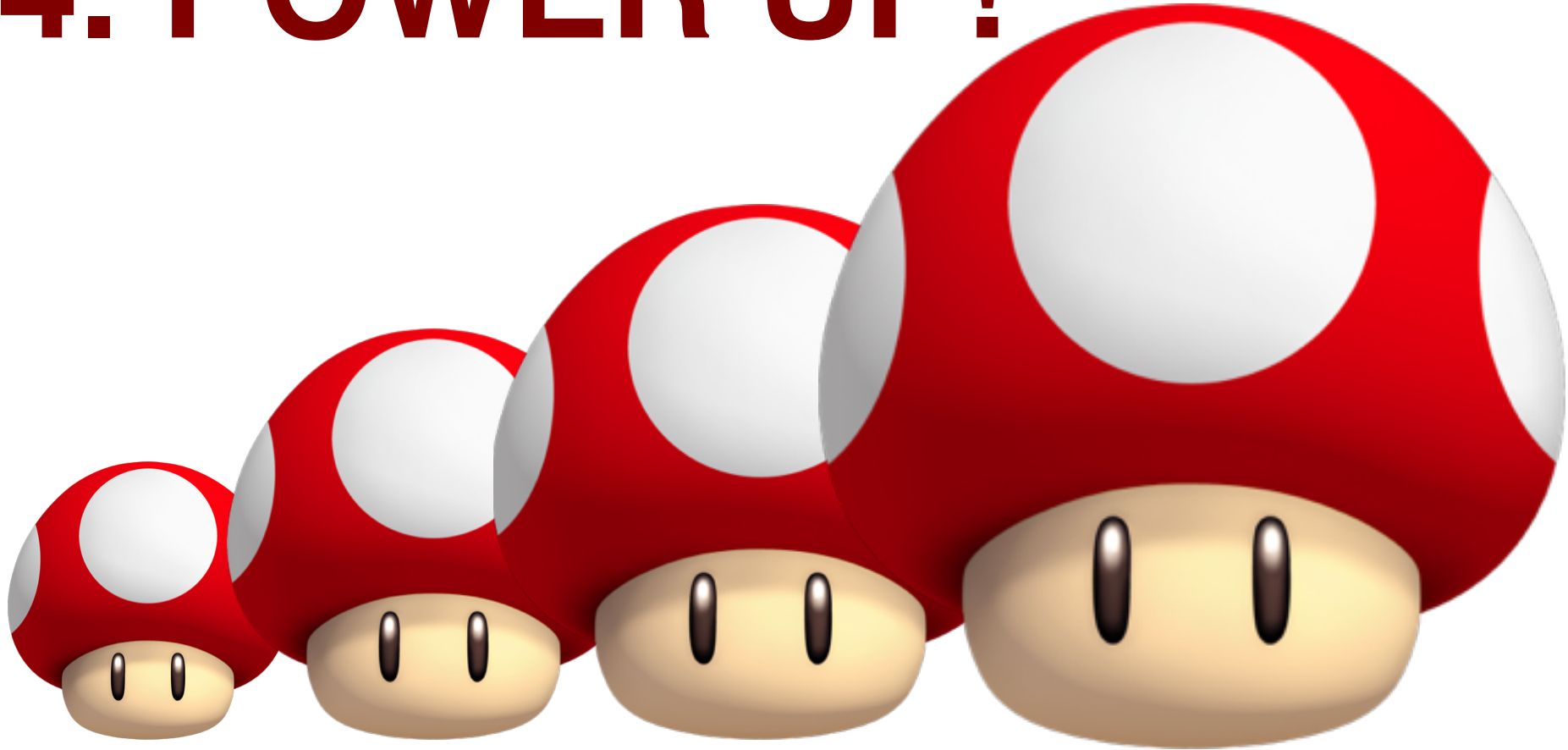
**DEEP, INTIMATE  
KNOWLEDGE OF YOUR  
ENVIRONMENT WILL  
SUPPORT YOUR TESTING**



# 3. UNDERSTAND YOUR WEAKNESSES



# 4. POWER UP!





# 5. USE TOOLS EFFECTIVELY





# KALI



# ZED ATTACK PROXY

The screenshot displays the OWASP ZAP interface during a scan of a WordPress site. The left sidebar shows the site structure with the following items:

- http://192.168.0.27
  - GET:wordpress
  - GET:wordpress(page\_id)
  - GET:wordpress(m)
  - GET:wordpress(cat)
  - wordpress
    - GET:wp-register.php
    - wp-admin
      - GET:wp-admin.css
      - images
        - GET:wordpress-logo.png
        - GET:fade-butt.png
    - POST:wp-register.php(action.submit.user\_email.user\_login)
    - GET:wp-login.php
    - POST:wp-login.php(log.pwd.redirect.to.submit)
    - GET:wp-login.php(action)
    - POST:wp-login.php(action.email.submit.user\_login)

The main pane shows the HTTP response for the selected resource. The status is 200 OK. The response body contains HTML code for a WordPress login page, including a title, meta tags, and a script tag.

The bottom pane shows the scan results table:

Id	Req. Timestamp	Method	URL	Code	Reason	RTT	Size Resp. ...	Highest Alert	Note	Tags
20	17/01/2014 10:36	POST	http://192.168.0.27/wordpress/wp-register.php	200	OK	218 ms	745 bytes	Low		
21	17/01/2014 10:43	GET	http://192.168.0.27/wordpress/wp-login.php	200	OK	47 ms	1.54 kB	Low		Form, Pa...
22	17/01/2014 10:49	POST	http://192.168.0.27/wordpress/wp-login.php	200	OK	63 ms	1.69 kB	Low		Form, Pa...
23	17/01/2014 10:51	GET	http://192.168.0.27/wordpress/wp-login.php?action=...	200	OK	44 ms	1.58 kB	Low		Form, m...
24	17/01/2014 10:01	POST	http://192.168.0.27/wordpress/wp-login.php	200	OK	87 ms	165 bytes	Low		
25	17/01/2014 10:02	GET	http://192.168.0.27/wordpress/wp-login.php	200	OK	51 ms	1.54 kB	Low		Form, Pa...
26	17/01/2014 10:04	GET	http://192.168.0.27/wordpress/	200	OK	132 ms	7.98 kB	Low		Form, Sc...
27	17/01/2014 10:14	GET	http://192.168.0.27/wordpress/	200	OK	59 ms	7.98 kB	Low		Form, Sc...





```
$ python sqlmap.py -u "http://target/vuln.php?id=1" --batch
```

```
sqlmap {1.0-dev-4512258}  
http://sqlmap.org
```

```
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual  
consent is illegal. It is the end user's responsibility to obey all applicable  
local, state and federal laws. Developers assume no liability and are not respon-  
sible for any misuse or damage caused by this program
```

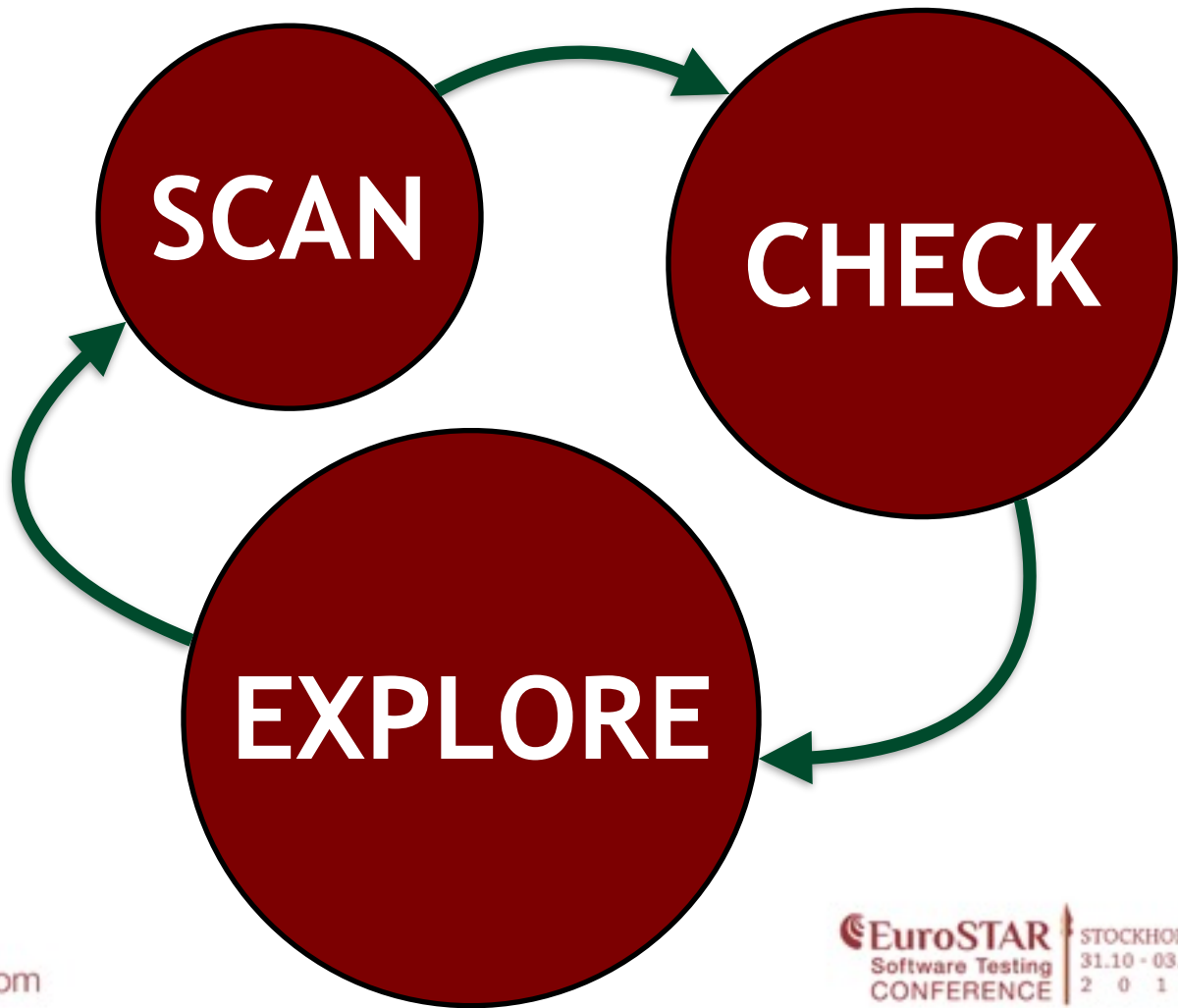
```
[*] starting at 15:02:07
```

```
[15:02:07] [INFO] testing connection to the target URL  
[15:02:07] [INFO] heuristics detected web page charset 'ascii'  
[15:02:07] [INFO] testing if the target URL is stable. This can take a couple of  
seconds  
[15:02:08] [INFO] target URL is stable  
[15:02:08] [INFO] testing if GET parameter 'id' is dynamic  
[15:02:08] [INFO] confirming that GET parameter 'id' is dynamic  
[15:02:08] [INFO] GET parameter 'id' is dynamic  
[15:02:08] [INFO] heuristic (basic) test shows that GET parameter 'id' might be  
injectable (possible DBMS: 'MySQL')
```

# SQLMAP



6.



A collage of ten images featuring various characters, mostly villains or antagonists, with the text "7. BE OCCASIONALLY EVIL" overlaid in large white letters. The characters include: a woman in a pink outfit (likely a Disney villainess), a character in a dark hood (likely a Disney villain), a character in a green hood (likely a Disney villain), a man with a beard (likely a Disney villain), a man in a suit (likely a Disney villain), a cartoon character on a motorcycle (likely a Disney villain), a man with a beard (likely a Disney villain), a man with a beard (likely a Disney villain), a man with a beard (likely a Disney villain), and a man with a beard (likely a Disney villain).

7. BE OCCASIONALLY EVIL

7. BE OCCASIONALLY EVIL

# As a hacker, I can

- **send bad data in URLs, so I can access data and functions for which I am not authorised**
- **send bad data in the content of requests...**
- **send bad data in HTTP headers...**
- **read and even modify all data that is input/output by your application**



# 8. DON'T DO IT ALONE







# 9. BE CLEAR, BE HEARD

# 10. BE DETERMINED



- 1. CONSIDER YOUR SCOPE**
- 2. KNOW YOUR STACK**
- 3. UNDERSTAND YOUR WEAKNESSES**
- 4. POWER UP!**
- 5. USE TOOLS EFFECTIVELY**
- 6. SCAN > CHECK > EXPLORE**
- 7. BE OCCASIONALLY EVIL**
- 8. DON'T DO IT ALONE**
- 9. BE CLEAR, BE HEARD**
- 10. BE DETERMINED**





# ANY QUESTIONS?

Daniel Billing  
Medidata, UK

@thetestdoctor

 **EuroSTAR**  
Software Testing  
CONFERENCE

STOCKHOLM  
31.10 - 03.11  
2 0 1 6

