

# A primer in software security testing

Software security is a hot topic. Users are losing their trust in software, especially when their machines are foreign beasts requiring continuous updates from multiple sources.



**Julian Harty**  
of Commercetest

## About the author

Julian is passionate about helping people to deliver reliable, fast systems that work. Over the last 18 years he has managed and maintained the performance of large-scale international online systems, and held roles including management of test and development teams. A special area of interest is to help improve testers' skills. Julian runs Commercetest, a company he founded in 1999

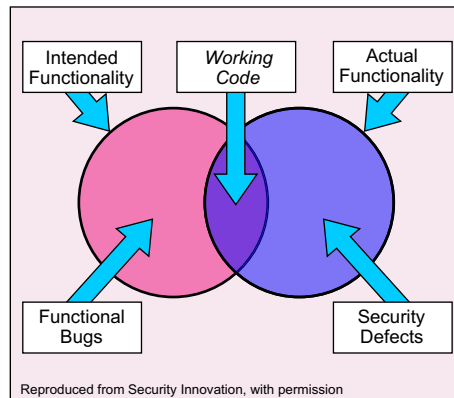


Fig.1 What's different about security testing?

**As an industry** we realise software needs to become more secure, but currently few of us are able to do meaningful security testing. This needs to change, and this article will help you to get started by providing practical advice on security testing.

## Why is security testing different / difficult?

Security testing sounds difficult, the preserve of translucent geeks who hack remote computer systems for the perverse fun of it, and little to do with average folk who have day jobs. Why do we think it's hard? Figure 1 shows that security testing

differs from functional testing as it concentrates on finding things that shouldn't occur, but do! [1]

Some aspects of security testing need advanced bit-twiddling skills, however there's plenty of scope for finding and preventing most security bugs without technical skills. For instance threat-modelling finds about 50% of security bugs according to Microsoft's Michael Howard [2] and it can be performed without technical skills. Threat-modelling can, and should, be performed early in the software development life cycle (SDLC), which helps us to prevent, or mitigate against, security bugs being created in the code.

## Security testing as a full SDLC activity

Penetration testing [3], while popular and necessary, is only practical once the software has been written and installed. Security testing should start from the beginning of the SDLC, and needs to consider the security requirements for the software and data from inception until after the system has been decommissioned. For instance hard drives have been sold on eBay that contained confidential and sensitive data. Threat modelling should have identified this as a risk and the company should have had procedures in place that ensured such data was securely erased before the system was decommissioned.

## Security testing techniques

So what sort of things can we do to test security? There are plenty of useful security testing techniques that help to trap security bugs. I will introduce various ones here, and Figure 2 shows where they fit in the SDLC.

## Threat modelling

Here we create models of the threats that violate the security constraints of a system and use these models to prevent or mitigate against those threats from affecting the

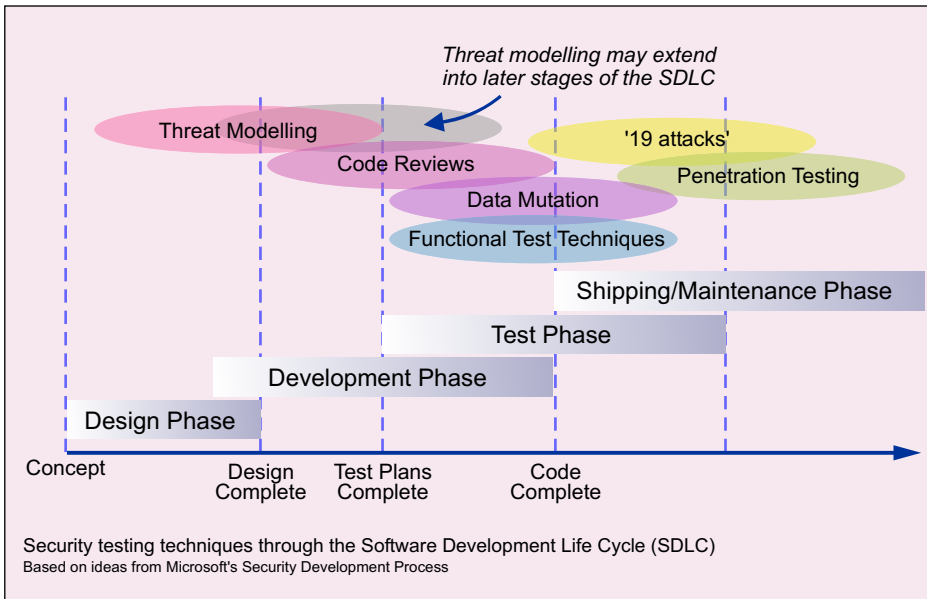


Fig. 2 Security-testing techniques through the SDLC

system. There are several useful modelling techniques:

1. Anti-goals: championed by Axel van Lamsweerde, a professor at Université Catholique de Louvain in Belgium. The model is rigorous; the material requires thought; however the ideas and concepts are worth learning and adapting to suit your needs [4].
2. Misuse cases: created by Guttorm Sindre and Andreas Opdahl, in 2000, and enhanced by Ian Alexander, 2001, these adapt the popular 'use cases' from UML to model attackers. Note that a similar model called abuse cases has been around since 1999 [5].
3. STRIDE: Microsoft's mnemonic that helps drive test techniques. A number of references are available including "Writing Secure Code 2nd edition" [6].

**Code reviews**

Many security bugs can be found by checking the source code, the process of doing so is called a *code review*. Code reviews can be performed by people, software tools, or a combination of both. The people need solid programming skills to be able to understand what the code does and to be able to find flaws in that code. Often the reviewers will be competent (ex-) programmers. Software tools scan the code quickly and try to identify patterns that indicate possible flaws. Each flaw can then be checked

manually to see if the flaw is a real vulnerability. Those that are need to be addressed, the rest are called false positives and effectively reduce the effectiveness of the tool because they distract the reader from real problems and take time to investigate.

There are both free and commercial code review tools available. For instance, [www.securesoftware.com](http://www.securesoftware.com) offers both RATS (free) and CodeAssure (commercial).

**Data mutation**

Michael Howard and David LeBlanc provide a great description of data mutation and explain how it can be used to attack software. They have identified 25 classes of mutation, which are grouped into 5 categories, as shown in Figure 3.

**Using special characters for SQL Injection**

We can test for SQL Injection vulnerabilities by using a combination of the special characters identified in the data mutation techniques. SQL is generally used to interrogate database servers: quotes are used around text values (**Cpq**), a semi-colon indicates the end of a command (**Cpm**), and the -- escape sequence is the start of a comment in the code (**Cpm**). Hackers quickly learnt they could modify and subvert the intended SQL by using a combination of these special characters, for instance to change:

```
SELECT FirstName, LastName, Balance
FROM Account
WHERE Login = 'UserLogin' AND
Password = 'UserPassword'
```

into:

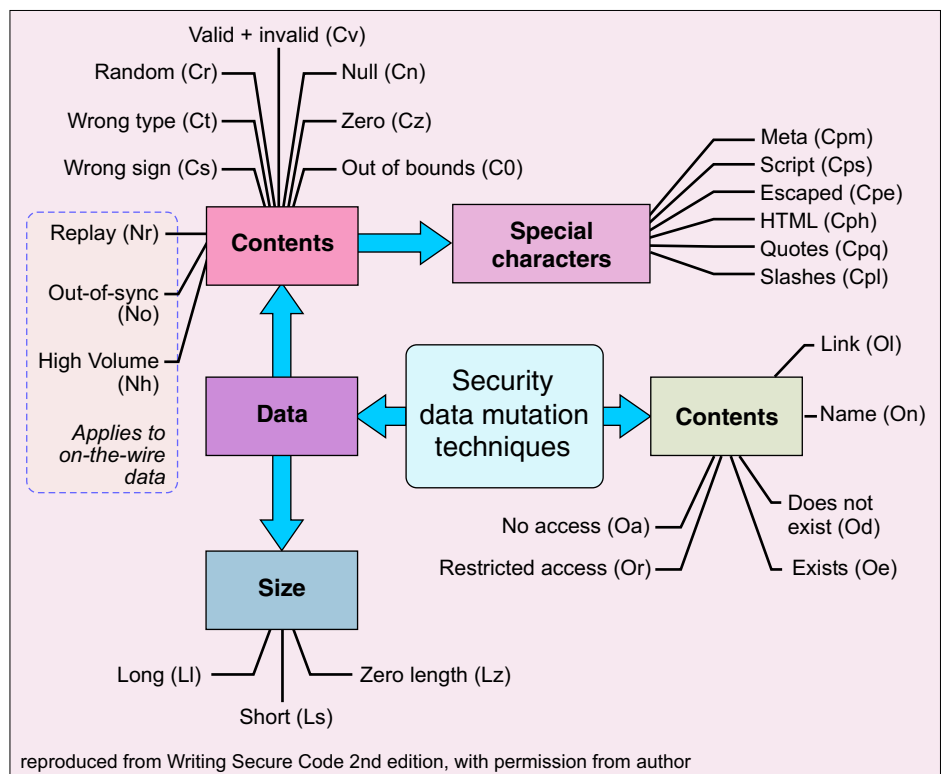


Fig. 3 Techniques to perturb applications to reveal security vulnerabilities and reliability bugs

```
SELECT FirstName, LastName, Balance
FROM Account
WHERE Login = ' ' OR 1=1; -- ' AND
Password = 'UserPassword'
```

where ' **OR 1=1; --** has been entered in the UserLogin input field. The single quote closes the immediately preceding open quote, the **OR 1=1** is always true which means the WHERE clause is true, and the **--** characters comment out the remaining code. This hack subverts the test for a valid username and password and causes the database to return all the rows from the database table.

### Functional testing techniques

Fifteen years ago Boris Beizer wrote in the second edition of 'Software Testing Techniques' [7] "They [malicious users] are out to get you. Some of them are programmers. They're persistent and systematic. .... And there are so many of them; so many of them and so few of us." He then explains how syntax-testing techniques help testers to find flaws in the validation logic for input fields, such as those we introduced in the SQL attack mentioned above.

Security testers can and should use existing functional testing techniques as part of their security testing. For instance, Boundary Value Analysis (BVA) exercises the boundary between valid and invalid partitions, for example, can I order -5 items from your web site? What happens: do I receive a credit? Common functional techniques are available free from the testing standards web site [8].

### 19 attacks

James Whittaker and Hugh Thompson produced an excellent and small book 'How to break software security' [9] that describes 19 attacks testers can use to test the security of software. The advice is very practical and you can apply the attacks immediately. All 19 attacks are described briefly on their web site [10].

Here is Attack 5 as an example:

- Force the application to operate in low memory, disk-space and network availability conditions.

By limiting the resources for a program the program may leak sensitive information, for example, to a temporary swap file, or otherwise untested, buggy, error handlers may be invoked.

### Penetration testing

Penetration testing is an established process where software is tested for known vulnerabilities to find out whether these, or similar vulnerabilities can be exploited.

### Getting started

The best place to start security testing is somewhere we feel safe and secure, where our mistakes will go unobserved and where any damage is limited. Luckily there are a number of useful free and commercial applications that can help us.

For instance the OWASP web site [11] offers WebGoat, a pedagogic web site where you can learn how to attack web sites in various ways. The site includes graded help that starts by giving you a clue, and ends by giving detailed instructions on how to find vulnerabilities. Experienced security testers might not need any help, while folk new to security testing can benefit from step-by-step instructions.

We can also use virtual machine emulators to expand the number of systems we can test against, while at the same time limiting our attacks to the physical machine we're using. Figure 4 contains four virtual machines on the right: ranging from an unpatched version of Windows 2000 Server, patched, and later versions of Windows Servers, and a Linux server running the open source Tomcat web server.

Virtual machines are easy to create and configure. Once they are ready you can run one or more simultaneously and try to do your worst to attack them. At the end of your attack you can choose whether to save the attacked, and possibly damaged, virtual server, or simply don't save the changes and start again.

Popular virtual machines are available from Microsoft, for example Microsoft Virtual PC [12] and from VMware [13].

There are a number of other commercial virtual machine programs available. For personal security testing I use the workstation versions of the respective products, as these are inexpensive and sufficient for my testing needs. I hope this brief article has encouraged you

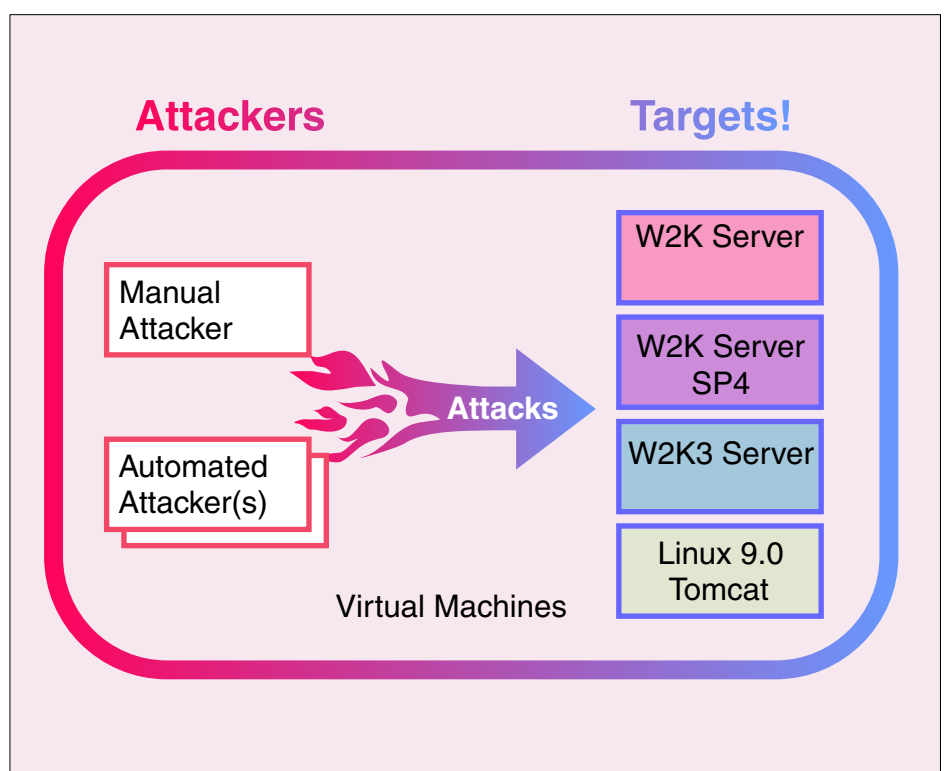


Fig. 4 Using virtual machines for security testing

## Further reading and references

1. <http://www.securityinnovation.com/chart.htm> - Why security bugs are different.
2. <http://www.cigital.com/ssw/presentations/howard.ppt> - Michael Howard's presentation where he mentions 50% of security bugs are found using threat analysis.
3. [http://en.wikipedia.org/wiki/Penetration\\_testing](http://en.wikipedia.org/wiki/Penetration_testing) - explains penetration testing.
4. <ftp://ftp.info.ucl.ac.be/pub/publi/2003/avl-RHAS03.pdf> - Scientific paper on Anti-goals.
5. <http://easyweb.easynet.co.uk/~iany/consultancy/papers.htm> Material on misuse cases, etc.
6. Writing Secure Code, second edition; Michael Howard and David LeBlanc; 2003 Chapter 19 on security testing is particularly useful, so is the rest of the book!
7. Software Testing Techniques, Second Edition; Boris Beizer; 1990 Still available second-hand and one of the standard references on software testing.
8. [http://www.testingstandards.co.uk/bs\\_7925-2.htm](http://www.testingstandards.co.uk/bs_7925-2.htm) - link to a document containing the common functional testing techniques.
9. How to Break Software Security; James Whittaker Herbert H. Thompson, 2004 a great, slim book with practical techniques for attacking software security.
10. <http://www.securityinnovation.com/resources/attacks/index.shtml> - Introduction to the 19 attacks.
11. <http://www.owasp.org> - great web site with free applications for web-based security testing, experimentation, and useful documents.
12. <http://www.microsoft.com/windows/virtualpc/default.aspx> - Home page for Microsoft's Virtual PC product, includes a 45-day free trial.
13. [http://www.vmware.com/products/desktop/ws\\_features.html](http://www.vmware.com/products/desktop/ws_features.html) - Home page for VMware's Workstation product. They offer a 30-day free trial.
14. <http://www.spidynamics.com/support/whitepapers/index.html> - very useful whitepapers on SQL Injection and other security issues and domains.
15. <http://www.ngssoftware.com/papers.htm> - more stimulating whitepapers.

to learn more about security testing, here are references that provide more information on the material introduced here ■

julian.harty@btinternet.com

**Subscribe to**  
Professional Tester Magazine

**Make a  
difference**

**We want you, if you:**

- > Are the best in your field
- > Have a passion for Testing
- > Have the ability to deliver the results that will help transform our clients' businesses.

For more information visit  
[www.newellandbudge.com/  
makeadifference](http://www.newellandbudge.com/makeadifference)

or email your CV to  
[testjobs@newellandbudge.com](mailto:testjobs@newellandbudge.com)  
(quoting ref PT3)

## The Fastest Growing Testing Company In The UK

...is looking for people with a passion for testing.



Newell & Budge is an established independent IT Consulting and Services business, that has been delivering excellence to our clients for over 20 years. Our specialist Testing Business is growing, as is our client base and reputation - as a result we are looking to expand our teams of talented testing professionals in London, Dublin, Belfast, Cheshire, Scotland, as well as the South East and South West of England.

With a reputation for innovation and thought leadership, Newell & Budge Testing provides professional, independent testing and validation services to some of Europe's largest organisations.

▶ **Test Delivery Managers** ▶ **Test Programme Managers**

▶ **Test Team Leaders** ▶ **Test Analysts** ▶ **Security Testing Specialists**

▶ **Test Engineers** ▶ **Tools Consultants**

We are growing fast and looking to expand in all locations. If you are an IT testing professional who wants to make a difference please get in touch.

**London-Horsham-Edinburgh-Glasgow-Birmingham-Dublin-Belfast-Delhi-Vancouver**