Bastian Ballmann

# Understanding Network Hacks

Attack and Defense with Python

Springer

# Understanding Network Hacks

Bastian Ballmann

# Understanding Network Hacks

Attack and Defense with Python

Springer

Bastian Ballmann
Uster, Switzerland

Translation from the German language edition "Network Hacks - Intensivkurs",
© Springer-Verlag, 2012

ISBN 978-3-662-44436-8 ISBN 978-3-662-44437-5 (eBook)
DOI 10.1007/978-3-662-44437-5
Springer Heidelberg New York Dordrecht London

Library of Congress Control Number: 2014960247

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

*For data travelers, knowledge hungry, curious, network-loving life-forms who like to explore and get to the bottom of thing.*

# Preface

Doesn't this book explain how to break into a computer system? Isn't that illegal and a bad thing at all?

I would like to answer both questions with no (at least the second one). Knowledge is never illegal nor something bad, but the things you do with it.

You as an admin, programmer, IT manager, or just an interested reader cannot protect yourself if you don't know the techniques of the attackers. You cannot test the effectiveness of your firewalls and intrusion detection systems or other security, related software if you are not able to see your IT infrastructure through the eyes of an attacker. You cannot weigh up the danger to costs of possible security solutions if you don't know the risks of a successful attack. Therefore, it is necessary to understand how attacks on computer networks really work.

The book presents a selection of possible attacks with short source code samples to demonstrate how easy and effectively and maybe undetected a network can be infiltrated. This way you can not only learn the real techniques but present them to your manager or employer and help them in the decision if it would make sense to care a little bit more about IT security. At the end of the book, you should be able to not only understand how attacks on computer networks really work but also to modify the examples to your own environment and your own needs.

Sure, the book also tells those bad guys how to crack the net and write their own tools, but IT security is a sword with two sharp blades. Both sides feed themselves off the same pot of knowledge, and it is a continuous battle, which the protecting side can never dream of winning if it censors itself or criminalizes their knowledge!

Uster, Switzerland Bastian Ballmann

# Contents

# Introduction

## Who Should Read This Book?

This book addresses interested Python programmers who want to learn about network coding and administrators who want to actively check the security of their systems and networks. The content should also be useful for white, gray, and black hat hackers, who prefer Python for coding, as well as for curious computer users, who want to get their hands on practical IT security and are interested in learning to see their network through the eyes of an attacker.

You neither need deep knowledge on how computer networks are built up nor in programming. You will get through all the knowledge you need to understand the source codes of the book in Chaps. 2 and 3. Readers, who know how to program in Python and dream in OSI layers or packet headers, can right away jump to Chap. 5 and start having fun at their device.

Of course a book like this needs a disclaimer, and the author would be happy if all readers only play on systems they are allowed to do so and use the information of this book only for good and ethical actions, otherwise, you may be breaking a law depending on the country your device is connected in.

The length of the book doesn't allow for in-depth discussion of all topics. You will only get somewhat more than the basics. If you want to dig deeper, you should afterward get some special lecture in your special field of interest.

## The Structure of the Book

The different hacks are grouped by network protocols, and every chapter content is ordered by difficulty. You can read the book in the order you like except both the introduction chapters about networks (Chap. 2) and Python (Chap. 3).

The code samples are printed unshortened; therefore, you can just copy and use them without worrying about incremental changes or add-ons. If you are too lazy or busy to type, you should consider downloading all sources by pointing for browsing software at http://www.codekid.net/pythonnetwork-hacks/all.zip.

At the end of each chapter, you will find a selection of tools also written in Python that attack the described protocol in a more detailed way.

Thanks to the basic knowledge learned in the chapter, it shouldn't be too hard to read and understand the source code of the tools.

## The Most Important Security Principles

The most important principles in building a secure network of the author's point of view are:

1. Security solutions should be simple. A firewall rule set that no one understands is a guarantee for security holes. Software that's complex has more bugs than simple code.
2. Less is more. More code, more systems, more services provide more possibilities of attack.
3. Security solutions should be open source. You can search easier for security problems if you have access to the source code. If the vendor disagrees to close an important security hole, you or someone else can fix it and you don't have to wait for six or more months till the next patch day. Proprietary software can have built-in backdoors sometimes called Law Interception Interface. Companies like Cisco (see RFC 3924), Skype (US-Patent-No 20110153809), and Microsoft (e.g., _NSAKEY http://en.wikipedia.org/wiki/NSAKEY) are only popular examples.
4. A firewall is a concept, not a box that you plug in and you are safe.
5. Keep all your systems up to date! A system that's considered secure today can be unprotected a few hours later. Update all systems, also smartphones, printer, and switches!
6. The weakest device defines the security of the complete system, and that doesn't necessarily have to be a computer; it can also be a human (read about social engineering).
7. There is no such thing as 100 % secure. Even a computer that is switched off can be infiltrated by a good social engineer. The aim should be to build that much layers that the attacker falls over one tripwire and leaves traces and that the value he or she can gain from a successful infiltration is much lower than the work or it kills his owner's skills.

# Chapter 1
# Installation

**Abstract** This chapter explains on which operating system the sources can be executed, which Python version you will need and how to install additional Python modules. Last but not least, we will discuss some possible solutions for setting up a complete development environment. If you are already familiar with the Python programming language you can skip this introductory chapter without missing anything.

## 1.1 The Right Operating System

Yes, I know the title of this section can lead to flame wars. It should just illustrate on which operating systems the source codes of this book are run. The author is using a GNU/Linux systems with kernel version 2.6.x and 3.x for development, but most of the sources, except the chapter about Bluetooth, should also runable on BSD or Mac OS X systems. If you succeed in running the source code on other systems the author would be happy if you could drop him a tiny email. Of course all other comments or criticisms are also welcome.

## 1.2 The Right Python Version

Python 3 has been released for quite a number of years now. However, we will nevertheless use Python 2.7, because nearly all modules we use are only available for this version of Python. Version 2.5 and 2.6 should also work but the author did not test it.

To check which version of Python is installed on your system, execute the following command

```
python --version
Python 2.7.2
```

If the output is less than 2.5 you should consider upgrading Python. If your version is 3.x think about installing Python 2.7 in parallel, but then you might have to change the interpreter path from /usr/bin/python to /usr/bin/python2 or /usr/bin/python2.7.

B. Ballmann, *Understanding Network Hacks*, DOI 10.1007/978-3-662-44437-5_1

## 1.3 Development Environment

The author prefers GNU/Emacs (www.gnu.org/software/emacs) as a development environment, because he thinks its editing and extension possibilities are unbeatable. Emacs supports all common features like syntax highlighting, code completion, code templates, debugger support, PyLint integration and thanks to Rope, Pymacs and Ropemacs, it has one of the best refactoring support for Python.

If you want to give Emacs and it features a try, the author suggests installing the awesome extension set Emacs-for-Python, downloadable at gabrielelanaro.github.com/emacs-for-python. Thanks to the amount of available plugins, Emacs can also be used as an email and Usenet client, for irc or jabber chatting, as music player and additional features like speech support, integrated shell and file explorer up to games like Tetris and Go. Some guys even think Emacs is not an IDE, but a whole operating system and use it as init process.

A good alternative for a console editor is Vim (www.vim.org) of course. The author does not like flame wars so if you do not know Emacs or Vim, give both a try. They are great! Vim includes all features of a modern IDE, is extensible and completely controllable with keyboard shortcuts and features a GUI version.

If you want to use one of those full-blown, modern IDEs, then check out Eclipse (www.eclipse.org) together with PyDev (pydev.org). Eclipse also has all the common features as well as code outlining, a better integrated debugging support and an endless seeming torrent of useful plugins like UMLet to draw UML diagrams or Mylyn to perfectly integrate a bugtracking system.

As alternative GUI-only IDE, you could also check out Eric4 (eric-ide.python-projects.org) and Spyder (code.google.com/p/spyderlib), which also include all common features plus a debugger, PyLint support and refactoring.

If you do not have that many resources and RAM for programming tasks, but need a GUI then Gedit might be the editor of your choice. However you should extend it with a bunch of plugins: Class Browser, External Tools, PyLint, Python Code Completion, Python Doc String Wizard, Python Outline, Source Code Comments and Rope Plugin.

The installation could be somewhat nasty and the functionality not as complete as for the other candidates. However, Gedit only uses the tenth of your RAM that Eclipse does.

The final choice is left to you. If you don't want to choose or try all possibilities, you should first try Eclipse with Pydev as bundle downloadable from Aptana (aptana.com/products/studio3). The chances are high that you will like it.

## 1.4 Python Modules

Python modules can be found in the Python packet index pypi.python.org. New modules can be installed by one of the following three possibilities:

1. Download the source archive, unpack it and execute the magic line

   ```
   python setup.py install
   ```

2. Use easy_install

   ```
   easy_install <modulname>
   ```

3. Get your feet wet with pip. Maybe you have to install a package like `python-pip` before you can use it.

   ```
   pip install <modulname>
   ```

You should use `pip`, because it also supports deinstallation and upgrading of one or all modules. You could also export a list of installed modules and its version, reinstall them on another system, you can search for modules and more.

Which Python modules are needed for which tools and source code snippets will be described at the beginning of the chapter or in the description of the snippet, if the module is only used for that code. This way, you will only install modules that you really want to use.

# Chapter 2
# Network 4 Newbies

**Abstract** Computer networks are the veins of the information age, protocols the language of the net.

This chapter describes the basics of networking starting with hardware going over to topology and the functionality of the most common protocols of an Ethernet/IP/TCP network up to Man-in-the-middle attacks. For all who want to rebuild or refresh their knowledge of networking.

## 2.1 Components

To be able to build a computer network of course you need some hardware. Depending on the kind of net you'll need cables, modems, old school acoustic in banana boxes, antennas or satellite receivers beside computers and network cards as well as router (Sect. 2.14), gateways (Sect. 2.13), firewalls Sect. 2.18, bridges (Sect. 2.15), hubs and switches.

A **hub** is just a simple box you plug network cables in and it will copy all signals to all connected ports. This property will probably lead to an explosion of network traffic. That's a reason why hubs are rarely used these days. Instead most of the time you will see **switches** building the heart of the network. The difference between a hub and a switch is a switch remembers the MAC address of the network card connected to the port and sends traffic only to the port it's destinated to. MAC addresses will be explained in more detail in Sect. 2.4.

## 2.2 Topologies

You can cable and construct computer networks in different ways. Nowadays the most common variant is the so called **star network** (see Fig. 2.1), where all computer are connected to a central device. The disadvantage is that this device is a single point of failure and the whole network will break down if it gets lost. This disadvantage can be circumstanced by using redundant (multiple) devices.

Another possibility is to connect all computers in one long row one after the other, the so called **bus network** (see Fig. 2.2). The disadvantage of this topology is that each computer must have two network cards and depending on the destination

B. Ballmann, *Understanding Network Hacks*, DOI 10.1007/978-3-662-44437-5_2