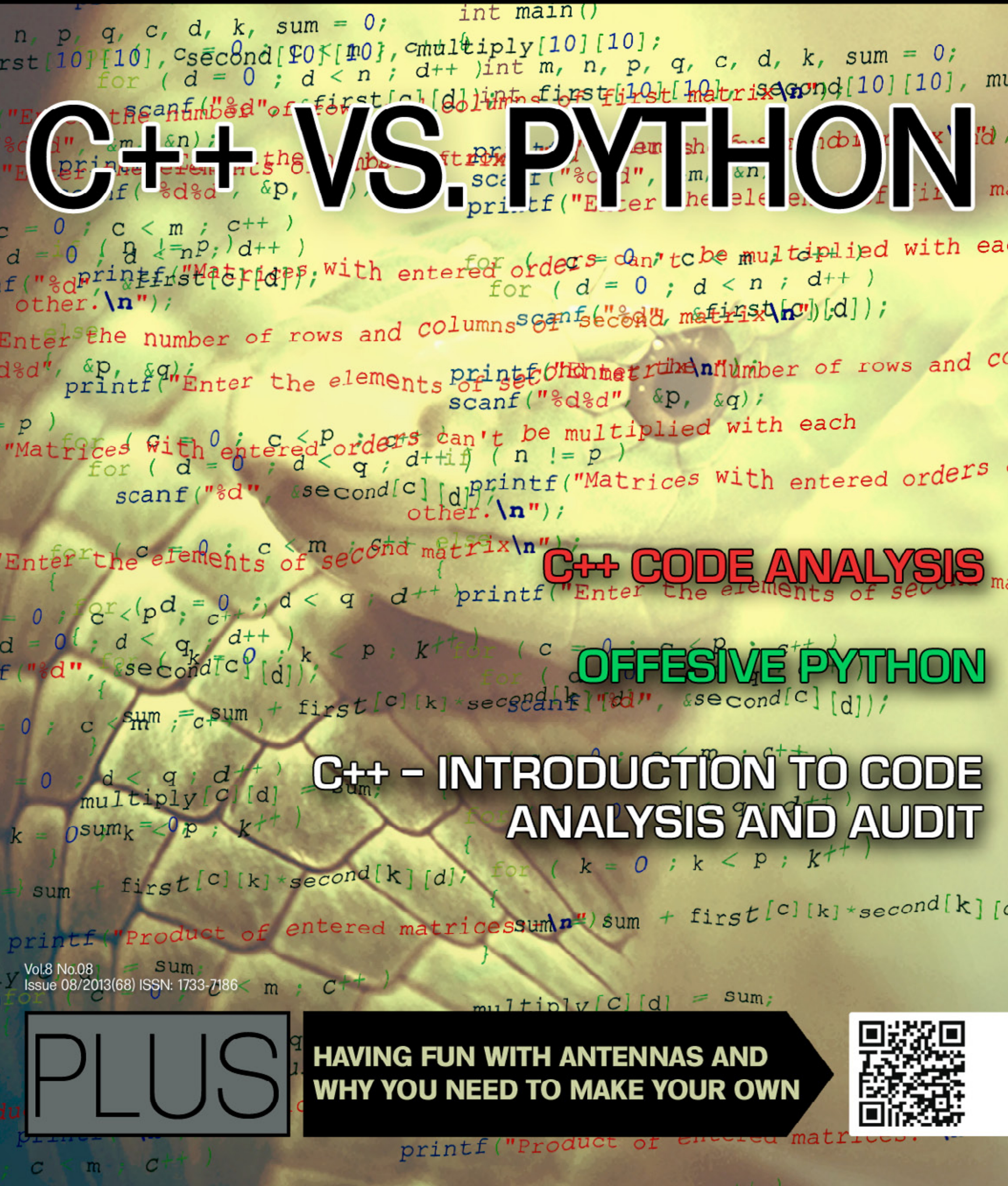
Hakin9

PRACTICAL PROTECTION

IT SECURITY MAGAZINE



C++ VS. PYTHON

C++ CODE ANALYSIS

OFFESIVE PYTHON

C++ – INTRODUCTION TO CODE ANALYSIS AND AUDIT

Vol.8 No.08
Issue 08/2013(68) ISSN: 1733-7186

PLUS

HAVING FUN WITH ANTENNAS AND WHY YOU NEED TO MAKE YOUR OWN

Joe Security LLC
Automated Malware Analysis

Next Generation Sandbox System

Joe Sandbox is an automated, highly configurable and scalable malware analysis system that provides extensive in-depth analysis reports to customers worldwide.

```
0x7c801a7c kern
el32.loadLibrar
yA(urlmon) 0x1a
474cce urlmon.U
RLDownloadToFil
eA(szURL=http:/
```

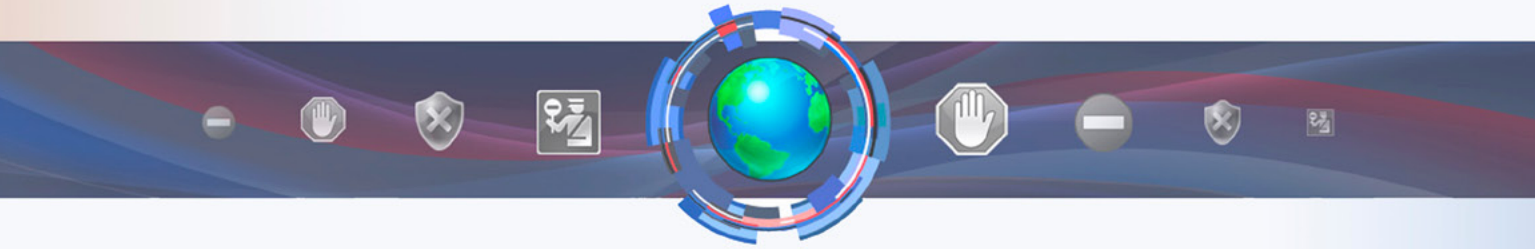
Technology Leader

Introducing **Hybrid Code Analysis**, Joe Security has developed a unique algorithm that combines dynamic and static code analysis in an intelligent way.

```
mov eax, 0047540Ch
ASCII "rtk64" move
eax, 0047541th ASC
II "rtk32" call dw
ord ptr [004570L8h
] CreateThread@KER
NEL32.DLL (Import,
4 Params)
```

Cross Platform

Joe Sandbox is the only fully-automated Sandbox System to support **Windows XP, Vista, W7, W7 x64 and Android** platforms.



Quality Support and Consulting

With direct access to the developer team, Joe Security provides excellent technical support and custom code to his customers.

JoeSecurity
www.joesecurity.org

Joe Security LLC
Hochbergerstrasse 60C
4057 Basel, Switzerland

swiss made software

Joe Security LLC

Automated Malware Analysis

Introducing Joe Sandbox Mobile!

The new solution for in-depth malware analysis on Android based systems.
Using **Hybrid Code Analysis**, static and dynamic analysis is combined in a clever way.

Powerful Instrumentation Engine

The highly-configurable, generic Instrumentation Engine not only analyzes **System API calls**, but any function matching specified signatures up to parameter level.

Generic Behavior Signatures

Providing an open interface and a solid initial set of generic behavior signatures, application activity is abstracted into well-formatted report data.

Free Services Available Online

All of Joe Security's Sandbox Systems are available as free web services at **apk-analyzer.net, file-analyzer.net, url-analyzer.net and document-analyzer.net**

JoeSecurity
www.joesecurity.org

Joe Security LLC
Hochbergerstrasse 60C
4057 Basel, Switzerland

swiss made software

Hakin9
team

Editor in Chief:
Ewelina Nazarczuk
ewelina.nazarczuk@hakin9.org

Editorial Advisory Board: Peter Harmsen, Dan Smith, Hans van Beek, Leighton Johnson, Gareth Watters, Sushil Verma, Jose Ruiz, Casey Parman, Wendy Bennington, Liew Edwin.

Proofreaders: Magdalena Gierwatowska, Krzysztof Samborski

Special thanks to our Beta testers and Proofreaders who helped us with this issue. Our magazine would not exist without your assistance and expertise.

Publisher: Paweł Marciniak

CEO: Ewa Dudzic
ewa.dudzic@hakin9.org

Product Manager:
Krzysztof Samborski
krzysztof.samborski@hakin9.org

Production Director:
Andrzej Kuca
andrzej.kuca@hakin9.org

Marketing Director:
Ewelina Nazarczuk
ewelina.nazarczuk@hakin9.org

Art Director: Ireneusz Pogroszewski
ireneusz.pogroszewski@software.com.pl
DTP: Ireneusz Pogroszewski

Publisher: Hakin9 Media sp. z o.o. SK
02-676 Warszawa, ul. Postępu 17D
Phone: 1 917 338 3631
www.hakin9.org

Whilst every effort has been made to ensure the highest quality of the magazine, the editors make no warranty, expressed or implied, concerning the results of the content's usage. All trademarks presented in the magazine were used for informative purposes only.

DISCLAIMER!

The techniques described in our articles may only be used in private, local networks. The editors hold no responsibility for misuse of the presented techniques or consequent data loss.

Dear Readers,

We would like to introduce a new issue made by Hakin9. This time you will deal with C++ Code and Python. You will learn how to conduct an audit with using C++ Code analysis. You can compare it with offensive programming with python. For sure after reading our step-by-step tutorials you will become a professional auditor. You will get to know how to analyze source code to find vulnerabilities which will help you to protect your websites and applications.

This time you will find a reach section Extra with articles about Payment Cards, Hardware Hacking and Evidence Analysis.

We hope you will like it!

Enjoy your time with Hakin9!

Regards,
Ewelina Nazarczuk
Hakin9 Magazine Junior Product Manager

and Hakin9 Team

◀ ▶

ADVANCED C++ CODE ANALYSIS

C++ – Introduction To Code Analysis and Audit 06

By Bamidele Ajayi
OCP, MCTS, MCITP EA, CISA, CISM

As a security professional code analysis and auditing is an essential task to unravel flaws and vulnerabilities. Analysis and auditing also sheds more light into what the code is actually doing.This article introduces you to the basics you need to know before embarking on source code audit and analysis with emphasis on C++.

C++ Code Analysis 10

By Mohmmed AlAbbadi, CISSP, CCSP, Deliver Meticulous Information Security Consultancy & Management Analysis for Decision Support, IS Influencer

Have you ever wanted to have a superpower? What was yours? The ability to fly? Blow fire? Disappear? Stop time or even go back in time? Run faster? Or be bulletproof? Mine was always the ability to scan objects and see what others couldn't see, the X-ray vision. Frankly, I wanted it for two reasons: one that was good and the other that was "wak". The first was to help people by finding (and sometimes fixing) problems-yet-to-happen-in-the-future before their manifestation. For example, scan a car to find out that the brakes don't work and tell the car owner before he/she drives it. The other reason was to find people's vulnerabilities (like a knee injury) to defend myself if I got attacked or bullied.

OFFENSIVE PYTHON

Offesive Python 16

By Kris Kaspersky, Reverse Engineering Expert, International Author

Python was created for fun, but evil hackers use it for profit. Why Python is a new threat for security industry and how tricky Lucifer's kids are – let's talk about it.

EXTRA

Having Fun With Antennas And Why You Need To Make Your Own 18

By Guillaume Puyo, Security researcher at EURA NOVA

Antennas (antennae for the serious people and entomologists) are the most omnipresent and the most misunderstood pieces of tech we all have, and yet, as everything keeps getting smaller and smaller they remain one of the few hacker friendly items we can tinker with. In these few pages, let's have a first basic approach on how they work, learn what's cool about them and get ready to build our own!

Payment Card Security 24

By Marios Andreou, Security Cosultant

There are many standards ensuring minimum level of protection to sensitive information such as the Payment Card Industry Data Security Standard (PCI DSS) which protects the cardholder's data (CHD), Data Protection Act, FSA regulations for financial information and ISO-27001 the information security Management standard. These standards are built to be more generic helping organisations deal with security risks and not to protect them from all security threats. Therefore, this article focuses on PCI DSS and what can be done and what approach must be followed by the experts to ensure security of information not just compliance with PCI DSS.

Evidence Analysis 28

By Mudit Sethia, Digital Evidence Analyst

Welcome back to the Novice approach to Evidence Analysis !! By putting the title to be one of being a novice, I really mean it to be novice – simple,straight and as it is. There can be no alteration done to the elementary alphabets ABCD ... Agreed ?? (btw I know the other 22 alphabets as well ;))

C++ - Introduction to Code Analysis And Audit

As a security professional code analysis and auditing is an essential task to unravel flaws and vulnerabilities. Analysis and auditing also sheds more light into what the code is actually doing. This article introduces you to the basics you need to know before embarking on source code audit and analysis with emphasis on C++.

Source Code audit and analysis is a comprehensive review which has a sole purpose of identifying bugs, flaws, and security breaches in software applications. It is a vital process which also attempts to unravel any violation in programming before software or application is released into production thereby reducing the attack surface. Code auditing and analysis has become the standard to ensure quality and security in software product. There are various ways to discovering vulnerabilities in systems or applications namely:

- Source Code Auditing and Analysis
- Reverse Engineering
- Fuzzing

Fuzzing is a software testing technique used in discovering flaws in coding and security loopholes in applications. This is technique is not limited to just applications, it could also be applicable to Operating Systems and Networking devices by sending or inputting large amounts of random data to the system to discover vulnerabilities. The application is monitored for any exceptions and a tool called fuzz tester can indicate potential causes.

Reverse Engineering is used to uncover features of an application which could reveal any vulnerability or security loopholes. This is process is very vital when there are no source code or documentation available for the application or system.

Source code auditing and analysis requires high knowledge and skills of any given programming language which in our case is C++. This process can be time consuming and tedious with an associated high cost when codes are not well documented (i.e. without comments) and convoluted.

Approaches

One of the very important rule of thumb for code audit and analysis is taking into consideration time constraints since we don't have the infinite luxury of time to audit and analyze the code. It is imperative to understand the product (application) written in the specific language which in our context is an application code snippet written in C++ with a clearly defined approach such as:

- Looking out for the most bugs
- Looking out for the easiest to find bugs
- Looking out for the weaknesses that are most reliable to exploit

With this clearly defined we can now prioritize our efforts. It is very important to limit the approach since we won't ever have enough time to find all the bugs.

Methodology

It is essential we have an understanding of the application. Such an understanding can be achieved with the following methods.

Reading specifications and documentation

Specifications and documentation helps in describing the minute detail of either all or specific parts of the application. This can also be akin to a functional specification. The documentation typically describes what is needed by the system user as well as requested properties of inputs and outputs. A functional specification is the more technical response onto a matching requirements document. Thus it picks up the results of the requirements analysis stage. On more complex systems multiple levels of functional specifications will typically nest to each other, e.g. on the system level, on the module level and on the level of technical details.

Understand purpose or business logic

Business logic refers to the underlying processes within a program that carry out the operations. Business logic is more properly thought of as the code that defines the database schema and the processes to be run, and contains the specific calculations or commands needed to carry out those processes. The user interface is what the customer sees and interacts with, while the business logic works behind the User Interface to carry out actions based on the inputted values.

Examining Attack Surface and Identify Target Components an Attacker Would Hit

The attack surface of a software environment is the code within a computer system that can be run by unauthorized users. This includes, but is not limited to: user input fields, protocols, interfaces, and services. OSSTMM 3 Defines Attack Surface as “The lack of specific separations and functional controls that exist for that vector”. One approach to improving information security is to reduce the attack surface of a system or software. By turning off unnecessary functionality, there are fewer security risks. By having less code available to unauthorized actors, there will tend to be fewer failures. Although attack surface reduction helps prevent security failures, it does not mitigate the amount of damage an attacker could inflict once vulnerability is found.

Source Code Analysis Tools

Source Code Analysis tools are designed to analyze source code and/or compiled version of code in order to help find security flaws. Ideally, such tools would automatically find security flaws with a high degree of confidence that what is found is indeed a flaw. However, this is beyond the state of the art for many types of application security flaws. Thus, such tools frequently serve as aids for an analyst to help them zero in on security relevant portions of code so they can find flaws more efficiently, rather than a tool that simply finds flaws automatically.

Strengths and Weaknesses of such tools

Strengths

Scales Well and Can be run on lots of software, and can be repeatedly.

For things that such tools can automatically find with high confidence, such as buffer overflows, SQL Injection Flaws, etc. they are great.

Weaknesses

Many types of security vulnerabilities are very difficult to find automatically, such as authentication problems, access control issues, insecure use of cryptography, etc. The current state of the art only allows such tools to automatically find a relatively small percentage of application security flaws. Tools of this type are getting better, however.

- High numbers of false positives.
- Frequently can’t find configuration issues, since they are not represented in the code.
- Difficult to ‘prove’ that an identified security issue is an actual vulnerability.

Many of these tools have difficulty analyzing code that can’t be compiled. Analysts frequently can’t compile code because they don’t have the right libraries, all the compilation instructions, all the code, etc.

Source Code Audit Tools

You can use different tools when conducting a source code audit. Below is a list of the most commonly used tools.

GrammaTech CodeSonar

CodeSonar is a source code analysis tool that performs a whole-program, interprocedural analysis on C and C++, and identifies programming bugs and security vulnerabilities at compile time. CodeSonar is used in the Defense/Aerospace, Medical, Industrial Control, Electronic, Telecom/Datacom and Transportation industries.

Splint

This tool is used to check programs developed in C for security vulnerabilities and coding mistakes.

Flawfinder

Flawfinder works by using a built-in database of well-known C and C++ function problems, such as buffer overflow risks, format string problems, race conditions, potential shell meta-character dangers, and poor random number acquisitions.

FindBugs

FindBugs uses static analysis to inspect code written in Java for occurrences of bug patterns and finds real errors in most Java software.

RATS

RATS, short for Rough Auditing Tool for Security, only performs a rough analysis of an application’s source code. It will not find all errors and may also flag false positives.

ITS4

This is a simple tool that statically scans C and C++ source code for potential security vulnerabilities. ITS4 is also a command-line tool that works across UNIX and Windows platforms by scanning source code and looking for function calls that are potentially dangerous.

GNU Visual Debugger

This debugger is licensed under the GNU Project and can be launched remotely via a variety of protocols, such as secured shells. The tool supports different languages, including C and C++.

Data Display Debugger

This graphical user interface debugger allows auditors to view an application's source code and display data structures. The tool supports debugging of many programming languages, such as C, C++, Java, Perl and other machine-level debugging.

Analysis and Audit Methods

Static Code Analysis

Static code analysis is the process of detecting errors and defects in software's source code. Static analysis can be viewed as an automated code review process. It deals with joint attentive reading of the source code and giving recommendations on how to improve it. This process reveals errors or code fragments that can become errors in future. It is also considered that the code's author should not give explanations on how a certain program part works. The program's execution algorithm should be clear directly from the program text and comments. If it is not so, the code needs improving.

Dynamic Program Analysis

Dynamic program analysis is the analysis of computer software that is performed by executing programs on a real or virtual processor. For dynamic program analysis to be effective, the target program must be executed with sufficient test inputs to produce interesting behavior. Use of software testing techniques such as code coverage helps ensure that an adequate slice of the program's set of possible behaviors has been observed.

Processing Results

The outcome of Code Analysis and Audit wouldn't be considered useful if the flaws of the application are not improved upon. The result should provide outcome so as to make recommendations on useful changes that needs to be implemented. This can only be achieved using complete documentation and accurate triaging

References

- *http://en.wikipedia.org/wiki/Functional_specification*
- *https://docs.google.com/presentation/d/16PiS_8oIzTwye58NsbRSipyNz9q-F-64eGZyalpbnLg/edit#slide=id.g79541baf_0_30*
- *http://en.wikipedia.org/wiki/Attack_surface*
- *http://en.wikipedia.org/wiki/Business_logic*
- *https://www.owasp.org/index.php/Source_Code_Analysis_Tools*
- *https://na.theiia.org/Pages/IIAHome.aspx*
- *http://en.wikipedia.org/wiki/Dynamic_code_analysis*
- *http://www.codeproject.com/Tips/344663/Static-code-analysis*

Documentation should include pointers to a flawed code, an explanation of the problem, and justification for why this is vulnerability. Adding recommendations for a fix is a useful practice but selecting and preparing the actual solution is the responsibility of the code owners.

Triaging process depends on the threshold of the security bug and also an understanding of the priorities. If the severity is set to high then immediate attention should be given for fixing.

Conclusion

C++ Code Analysis and Audit provides useful information on security vulnerabilities and recommendations for redesign. It also provides opportunity for organizational awareness which would improve effectiveness and help to prioritize efforts.

Automated security tools are able to identify more errors but some vulnerabilities might be missed. Manual analysis shouldn't be a replacement for these tried and tested tools, but it can be advantageously integrated with them.

Manual analysis is more expensive, difficult, and highly dependent on the experience of who is doing the analysis and audit. However, in many situations this investment is worthwhile to obtain acceptable level of confidence.

BAMIDELE AJAYI

(CISM, CISA, OCP, MCTS, MCITP EA) is an Enterprise Systems nEngineer experienced in planning, designing, implementing and madministering LINUX and WINDOWS mbased systems, HA cluster Databases and Systems, SAN and Enterprise Storage Solutions. Incisive and highly dynamic Information Systems Security Personnel with vast security architecture technical experience devising, integrating and successfully developing security solutions across multiple resources, services and products.

SW GLOBAL

SW Global's eGovernment solutions automate and enable the virtual flow of public services and information, and promote governance and public participation within public institutions.

Our eGovernment solutions are based on a comprehensive model called "eChange," which provides innovative approaches and new technologies to accelerate growth through the institutional adoption of reform.

Our Company's solutions are underlined by a broader social context; they aim to drive the transformation of under-developed societies by using technology to increase efficiency and effectiveness within the public sector.

Our company will act as a vehicle for promoting technology as a tool for development in a global society where knowledge is the currency of exchange.

SW Global is one of the first globally competitive technology companies to emerge from West Africa and penetrate the global information technology (IT) market.

SW Global has many platforms to support this effort, namely:

Driver and Vehicle Licensing Solution (DVLS)
Promotes national safety and facilitates road user administration through the management of license, automobile, and driver information

Integrated Postal Management System (IPMS)
Automates the core processes of a postal enterprise, managing various types of mail, mail related transactions, and electronic money transfer

Municipality Management System (MMS)
Adapts to the various needs of municipal government, offering services ranging from tax collection to human resource management to eProcurement

National Immigration Management System (NIMS)
Strengthens immigration controls by centralizing passport and visa applications processes, border control, and the issuance of work permits

National Pension Management System (NPMS)
Provides a secure platform for automated pension management and funds transfer, including surveillance and regulation features

SIM Subscriber Platform
Technology basis for an initiative to register SIM card information in Africa. Meets government security needs and offers many value adds to cell phone users.

A DREXEL COMPANY

C++ Code Analysis

Have you ever wanted to have a superpower? What was yours? The ability to fly? Blow fire? Disappear? Stop time or even go back in time? Run faster? Or be bulletproof? Mine was always the ability to scan objects and see what others couldn't see, the X-ray vision. Frankly, I wanted it for two reasons: one that was good and the other that was "wak".

The first was to help people by finding (and sometimes fixing) problems-yet-to-happen-in-the-future before their manifestation. For example, scan a car to find out that the brakes don't work and tell the car owner before he/she drives it. The other reason was to find people's vulnerabilities (like a knee injury) to defend myself if I got attacked or bullied.

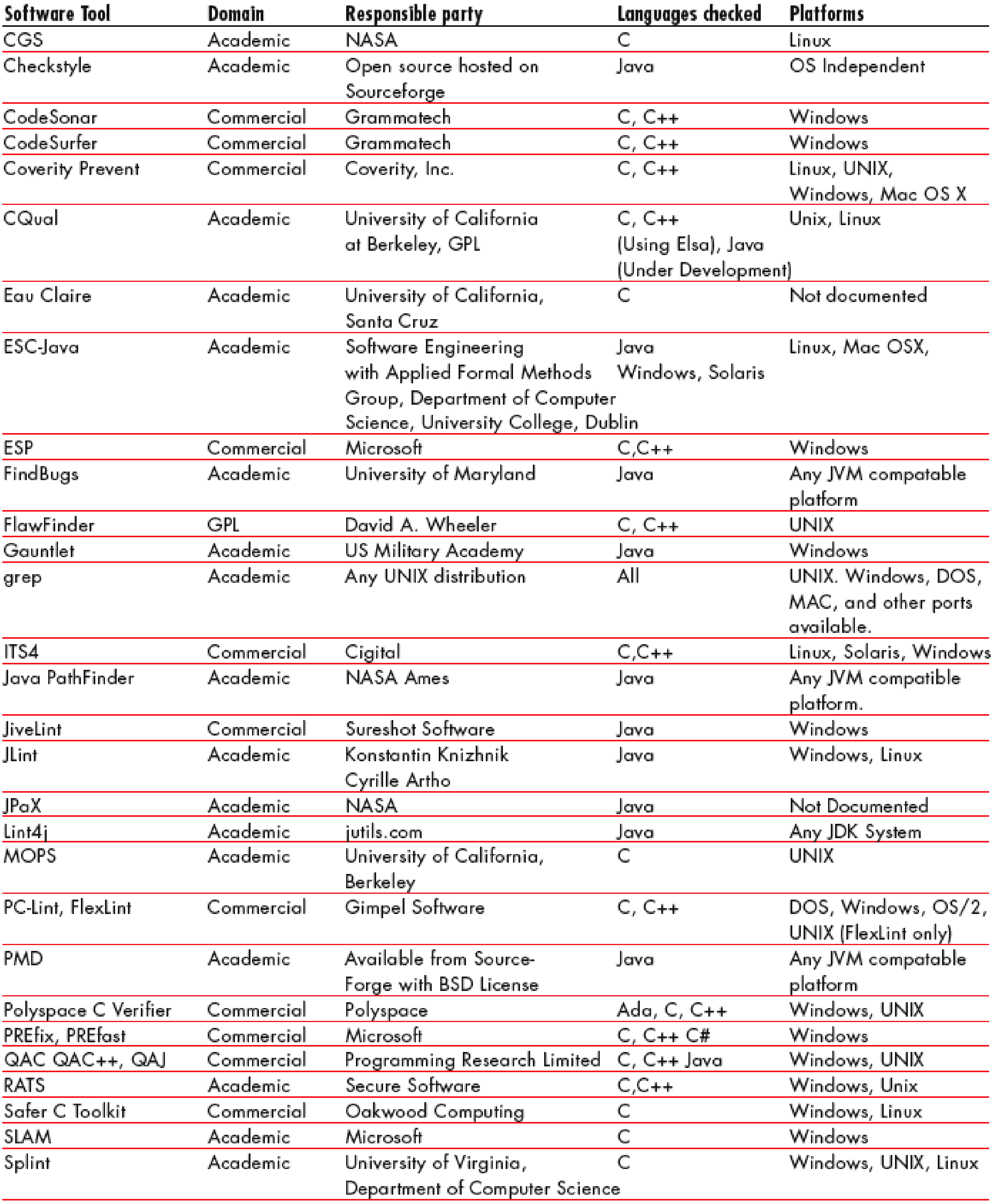
Figure 1. *Superman*

Though it used to be a dream as a child, today it is a reality; not only for me but for all of us (at least, the vigilante wannabes). Now, hold your horses; this is neither another scientific breakthrough about a new technology that perhaps can be integrated with Google Glass (I wish it were), nor a "limitless" magical pill that mutates your eye structure. This is simply the ability to scan the "core" of almost all objects around us to find their vulnerabilities and correct them ahead of time before someone else finds them and exploits them. Did I say the "core"?!! Ooh, I meant the "code". Confused yet? Let me explain [1].

Most systems today are computerized (hint: the car mentioned above) and therefore they are basically pieces of "code". The so called "code" is developed by programmers in different programming languages (such as Java, .Net, C++, etc) and may include weaknesses or vulnerabilities that allow people, like me a s a child, to abuse them (hint: injecting code into the car to change its behavior). On the other hand, sometimes you may have access to this original readable code in its raw text format (or what we call, the source code), and some other times you may need to "extract", derive or guess (reverse engineer) the original code from a running code that is unreadable. In either case, once you have access to the source or derived code, you will be able to put on the "Dev-man" vigilante suit and unleash the x-ray vision superpower by analyzing the code for issues using ready-made tools, most of which are free, that are available on the Internet.

For the sake of the rest of this article, all the analysis, tools and code examples will be related to C++.

Unlike Smallville, you don't have to wait four episodes to get familiar with the X-ray vision; you'll get it all here. In this article we'll go through the whole vicious cycle from a risk management perspective (not in detail). We'll start with explaining the root cause of the problem, identifying the risk



Software Tool	Domain	Responsible party	Languages checked	Platforms
CGS	Academic	NASA	C	Linux
Checkstyle	Academic	Open source hosted on Sourceforge	Java	OS Independent
CodeSonar	Commercial	Grammatech	C, C++	Windows
CodeSurfer	Commercial	Grammatech	C, C++	Windows
Coverity Prevent	Commercial	Coverity, Inc.	C, C++	Linux, UNIX, Windows, Mac OS X
CQual	Academic	University of California at Berkeley, GPL	C, C++ (Using Elsa), Java (Under Development)	Unix, Linux
Eau Claire	Academic	University of California, Santa Cruz	C	Not documented
ESC-Java	Academic	Software Engineering with Applied Formal Methods Group, Department of Computer Science, University College, Dublin	Java Windows, Solaris	Linux, Mac OSX,
ESP	Commercial	Microsoft	C,C++	Windows
FindBugs	Academic	University of Maryland	Java	Any JVM compatable platform
FlawFinder	GPL	David A. Wheeler	C, C++	UNIX
Gauntlet	Academic	US Military Academy	Java	Windows
grep	Academic	Any UNIX distribution	All	UNIX. Windows, DOS, MAC, and other ports available.
ITS4	Commercial	Cigital	C,C++	Linux, Solaris, Windows
Java PathFinder	Academic	NASA Ames	Java	Any JVM compatible platform.
JiveLint	Commercial	Sureshot Software	Java	Windows
JLint	Academic	Konstantin Knizhnik Cyrille Artho	Java	Windows, Linux
JPaX	Academic	NASA	Java	Not Documented
Lint4j	Academic	jutils.com	Java	Any JDK System
MOPS	Academic	University of California, Berkeley	C	UNIX
PC-Lint, FlexLint	Commercial	Gimpel Software	C, C++	DOS, Windows, OS/2, UNIX (FlexLint only)
PMD	Academic	Available from Source-Forge with BSD License	Java	Any JVM compatable platform
Polyspace C Verifier	Commercial	Polyspace	Ada, C, C++	Windows, UNIX
PREfix, PREfast	Commercial	Microsoft	C, C++ C#	Windows
QAC QAC++, QAJ	Commercial	Programming Research Limited	C, C++ Java	Windows, UNIX
RATS	Academic	Secure Software	C,C++	Windows, Unix
Safer C Toolkit	Commercial	Oakwood Computing	C	Windows, Linux
SLAM	Academic	Microsoft	C	Windows
Splint	Academic	University of Virginia, Department of Computer Science	C	Windows, UNIX, Linux

Table 1. *Summary of static-analysis tools*

(by exploring the threat, the vulnerability & the impact), mitigating the risk and monitoring the effectiveness of the mitigation strategy.

The problem in a nutshell

Developers code in different languages and each language has its own quality, performance & security issues. If C/C++ is the chosen language, then you're prone to security issues just as much. In addition to issues (such as authentication problems and access control issues) that are difficult to find using automated tools, security issues that are related to memory mishandling/mismanagement that is referred to as Buffer Overflow are easier to spot. Simply put, the problem is that if an attacker managed to exploit a system that is vulnerable to buffer overflows, the impact will be massive (depending on the criticality of the system). The impact is realized in either the denial of the system service (corruption of valid data, system halt, restart or crash) or the execution of the attacker's system-injected code (escalating privileges & spitting out the system password are good starters).

The solution in a nutshell

Yes, you probably guessed it right. Code analysis it is. However, in organizations the solution involves more than the static code analysis. In addition to incorporating the static code review within the software development cycle, the continuous process of identifying security risks, mitigating them, monitoring the mitigation effectiveness and governing the whole process is the way to go.

Where to start?

The idea is to use a tool to automatically detect all security flaws and recommend corrections. There are different types of tools that can be used in different situations. If the source code is available, then static code analysis tools are used to detect flaws. Otherwise, debuggers/disassemblers can be used to reverse engineer the compiled code and identify buffer overflows. Fuzzing techniques and tools can be used to provide random or invalid data input to applications to observe their behavior. Having said all of that, a simple text editor like notepad is sufficient to manually review the code, but it takes more time, effort and knowledge. In Table 1, you'll find examples of famous static code analysis Tools [2].

What are we trying to find?

To answer this question, we first need to explain the anatomy of a buffer and then show how things can go wrong. Consider the code below [3]:

```
#include <iostream>
int main(){

    char A[8];
    int n=0;
    cout << "Please enter a word\n";
    while (cin >> A[n]) {
    n++;

    }
    return 0;
}
```

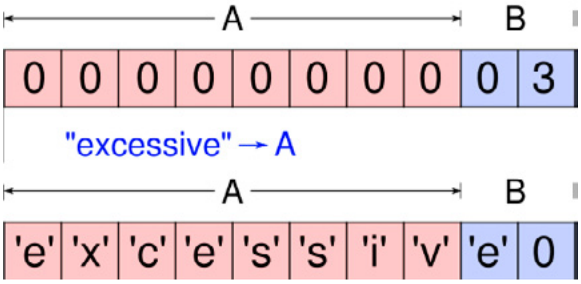


Figure 2. *Anatomy of a buffer*

The problem with the above code is that when the program asks the user to enter a word, it doesn't check the array boundaries. Though "A" is of 8-character size, putting a 9-character word such as "excessive" as an input will overflow the allocated 8 character A buffer and overwrite the B buffer with the "e" character and the null character.

Buffer overflows are generally of many types: Stack based and Heap based are a few and fall under – but not limited to – one or more of the following categories:

- Boundary Checking (like the example above)
- String format
- Constructors & Destructors
- Use-after-free
- Type confusion
- Reference pointer

The objective of the article is not to explore all types of buffer overflows and code review techniques rather an overview of the whole process.

Detection/identification tools

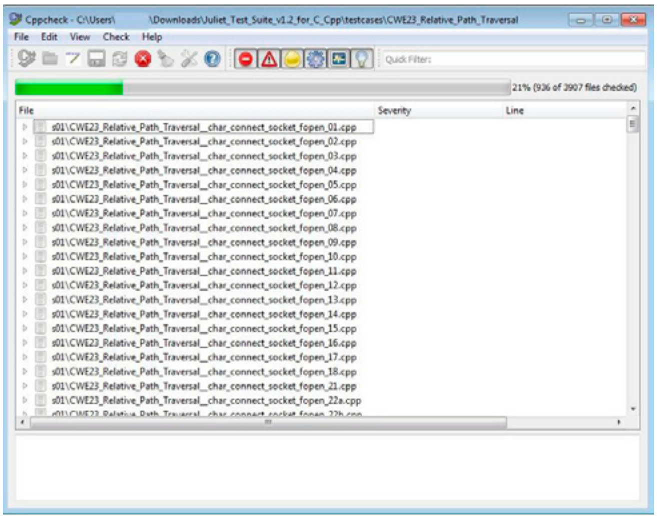
There are many static code analysis tools, some of which are commercial such as IBM Appscan Source Edition and HP Fortify Static Code Analyzer, and some of which are academic/free/open source such as Flawfinder, Clang Static Analyzer and Cppcheck. Below is a snapshot of Cppcheck under progress. Notice that more than 3000 code files got to be analyzed in under one minute (Figure 3).

The next snapshot shows the results of the analysis (Figure 4).

A Static code analysis example

The National Institute of Standards and Technology NIST Software Assurance Metrics And Tool Evaluation SAMATE project “is sponsored by the U.S. Department of Homeland Security (DHS) National Cyber Security Division and NIST” [4]. The project offers test suites for public download. The test suites contain C/C++ files that are vulnerable to different types of attacks. The project goes the extra mile and includes the solution to each test case within the same file. Let’s take one example of the test suites and explain the vulnerability and the solution.

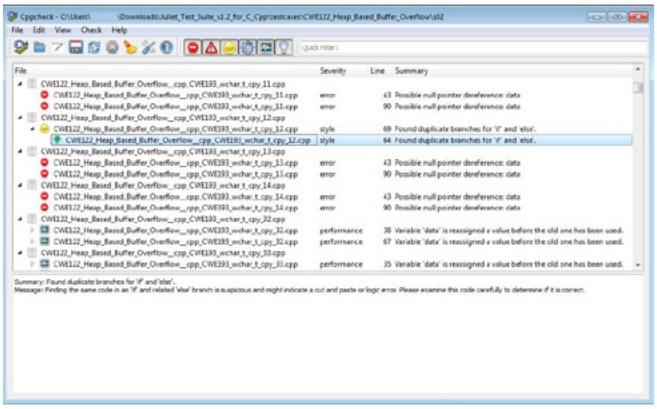
“Targeted at both the development community and the community of security practitioners, Common Weakness Enumeration (CWE™) is a formal list or dictionary of common software weaknesses that can occur in software’s architecture, design, code or implementation that can lead to exploitable security vulnerabilities” [2]. According to CWE 805 that is titled “Buffer Access with Incorrect Length Value”, “the software uses a sequential operation to read or write a buffer, but it uses an incorrect length value” (look at the figure below) “that causes it to access memory that is outside of the bounds of the buffer.” [6]


Cppcheck - C:\Users\ \Downloads\Juliet_Test_Suite_v1.2_for_C_Cpp\testcases\CWE23_Relative_Path_Traversal
File Edit View Check Help
Quick Filter:

```
21% (936 of 3907 files checked)

File                                                              Severity     Line
s01\CWE23_Relative_Path_Traversal__char_connect_socket_fopen_01.cpp
s01\CWE23_Relative_Path_Traversal__char_connect_socket_fopen_02.cpp
s01\CWE23_Relative_Path_Traversal__char_connect_socket_fopen_03.cpp
s01\CWE23_Relative_Path_Traversal__char_connect_socket_fopen_04.cpp
s01\CWE23_Relative_Path_Traversal__char_connect_socket_fopen_05.cpp
s01\CWE23_Relative_Path_Traversal__char_connect_socket_fopen_06.cpp
s01\CWE23_Relative_Path_Traversal__char_connect_socket_fopen_07.cpp
s01\CWE23_Relative_Path_Traversal__char_connect_socket_fopen_08.cpp
s01\CWE23_Relative_Path_Traversal__char_connect_socket_fopen_09.cpp
s01\CWE23_Relative_Path_Traversal__char_connect_socket_fopen_10.cpp
s01\CWE23_Relative_Path_Traversal__char_connect_socket_fopen_11.cpp
s01\CWE23_Relative_Path_Traversal__char_connect_socket_fopen_12.cpp
s01\CWE23_Relative_Path_Traversal__char_connect_socket_fopen_13.cpp
s01\CWE23_Relative_Path_Traversal__char_connect_socket_fopen_14.cpp
s01\CWE23_Relative_Path_Traversal__char_connect_socket_fopen_15.cpp
s01\CWE23_Relative_Path_Traversal__char_connect_socket_fopen_16.cpp
s01\CWE23_Relative_Path_Traversal__char_connect_socket_fopen_17.cpp
s01\CWE23_Relative_Path_Traversal__char_connect_socket_fopen_18.cpp
s01\CWE23_Relative_Path_Traversal__char_connect_socket_fopen_21.cpp
s01\CWE23_Relative_Path_Traversal__char_connect_socket_fopen_22a.cpp
s01\CWE23_Relative_Path_Traversal__char_connect_socket_fopen_22b.cpp
```

Figure 3. *Cppcheck under progress*

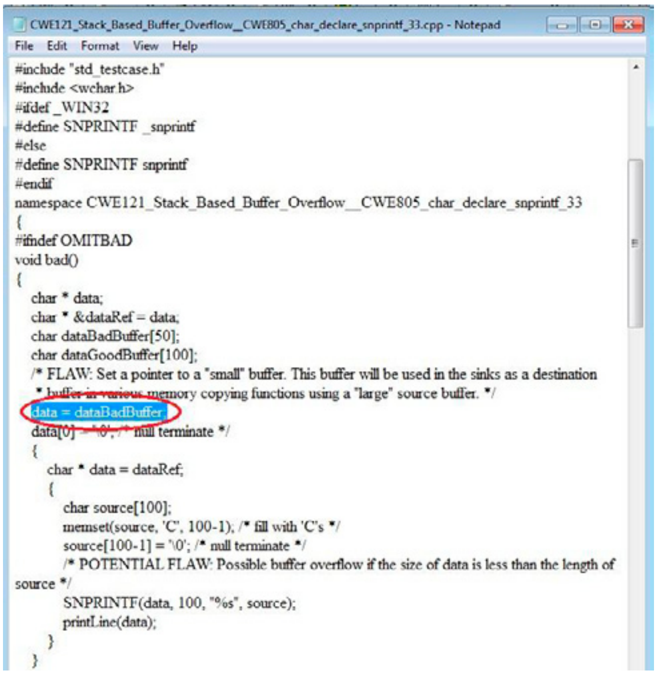

Cppcheck - C:\Users\ \Downloads\Juliet_Test_Suite_v1.2_for_C_Cpp\testcases\CWE122_Heap_Based_Buffer_Overflow\s02
File Edit View Check Help
Quick Filter:

```
File                                                              Severity     Line  Summary
CWE122_Heap_Based_Buffer_Overflow__cpp_CWE193_wchar_t_cpy_11.cpp
  CWE122_Heap_Based_Buffer_Overflow__cpp_CWE193_wchar_t_cpy_11.cpp  error         43  Possible null pointer dereference: data
  CWE122_Heap_Based_Buffer_Overflow__cpp_CWE193_wchar_t_cpy_11.cpp  error         90  Possible null pointer dereference: data
CWE122_Heap_Based_Buffer_Overflow__cpp_CWE193_wchar_t_cpy_12.cpp
  CWE122_Heap_Based_Buffer_Overflow__cpp_CWE193_wchar_t_cpy_12.cpp  style         69  Found duplicate branches for 'if' and 'else'.
    CWE122_Heap_Based_Buffer_Overflow__cpp_CWE193_wchar_t_cpy_12.cpp style        64  Found duplicate branches for 'if' and 'else'.
CWE122_Heap_Based_Buffer_Overflow__cpp_CWE193_wchar_t_cpy_13.cpp
  CWE122_Heap_Based_Buffer_Overflow__cpp_CWE193_wchar_t_cpy_13.cpp  error         43  Possible null pointer dereference: data
  CWE122_Heap_Based_Buffer_Overflow__cpp_CWE193_wchar_t_cpy_13.cpp  error         90  Possible null pointer dereference: data
CWE122_Heap_Based_Buffer_Overflow__cpp_CWE193_wchar_t_cpy_14.cpp
  CWE122_Heap_Based_Buffer_Overflow__cpp_CWE193_wchar_t_cpy_14.cpp  error         43  Possible null pointer dereference: data
  CWE122_Heap_Based_Buffer_Overflow__cpp_CWE193_wchar_t_cpy_14.cpp  error         90  Possible null pointer dereference: data
CWE122_Heap_Based_Buffer_Overflow__cpp_CWE193_wchar_t_cpy_32.cpp
  CWE122_Heap_Based_Buffer_Overflow__cpp_CWE193_wchar_t_cpy_32.cpp  performance   38  Variable 'data' is reassigned a value before the old one has been used.
  CWE122_Heap_Based_Buffer_Overflow__cpp_CWE193_wchar_t_cpy_32.cpp  performance   67  Variable 'data' is reassigned a value before the old one has been used.
CWE122_Heap_Based_Buffer_Overflow__cpp_CWE193_wchar_t_cpy_33.cpp
  CWE122_Heap_Based_Buffer_Overflow__cpp_CWE193_wchar_t_cpy_33.cpp  performance   35  Variable 'data' is reassigned a value before the old one has been used.

Summary: Found duplicate branches for 'if' and 'else'.
Message: Finding the same code in an 'if' and related 'else' branch is suspicious and might indicate a cut and paste or logic error. Please examine this code carefully to determine if it is correct.
```

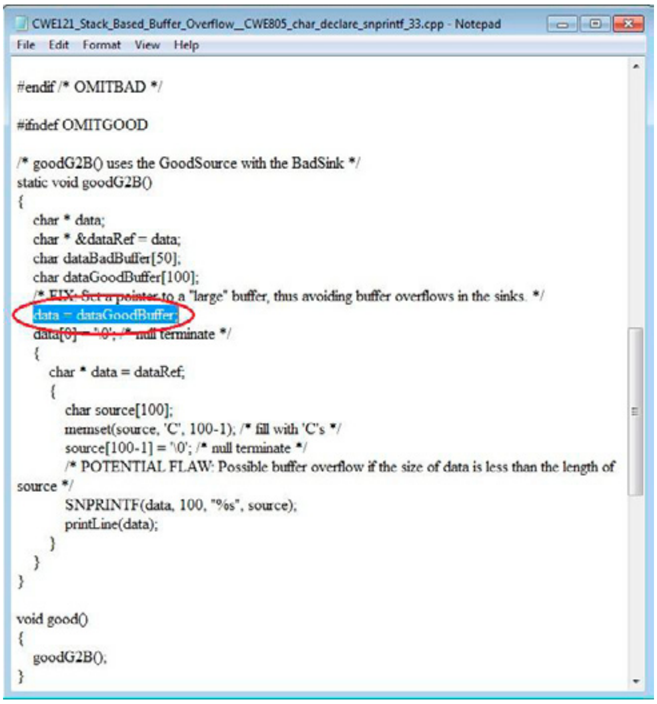
Figure 4. *The results of the analysis*

And the solution lies in setting the pointer to a large buffer as illustrated Figure 6.


CWE121_Stack_Based_Buffer_Overflow__CWE805_char_declare_snprintf_33.cpp - Notepad
File Edit Format View Help

```
#include "std_testcase.h"
#include <wchar.h>
#ifdef _WIN32
#define SNPRINTF _snprintf
#else
#define SNPRINTF snprintf
#endif
namespace CWE121_Stack_Based_Buffer_Overflow__CWE805_char_declare_snprintf_33
{
#ifndef OMITBAD
void bad()
{
  char * data;
  char * &dataRef = data;
  char dataBadBuffer[50];
  char dataGoodBuffer[100];
  /* FLAW: Set a pointer to a "small" buffer. This buffer will be used in the sinks as a destination
   * buffer in various memory copying functions using a "large" source buffer. */
  data = dataBadBuffer;
  data[0] = '\0'; /* null terminate */
  {
    char * data = dataRef;
    {
      char source[100];
      memset(source, 'C', 100-1); /* fill with 'C's */
      source[100-1] = '\0'; /* null terminate */
      /* POTENTIAL FLAW: Possible buffer overflow if the size of data is less than the length of
source */
      SNPRINTF(data, 100, "%s", source);
      printLine(data);
    }
  }
}
```

Figure 5. *The software uses a sequential operation to read or write a buffer, but it uses an incorrect length value*


CWE121_Stack_Based_Buffer_Overflow__CWE805_char_declare_snprintf_33.cpp - Notepad
File Edit Format View Help

```
#endif /* OMITBAD */

#ifndef OMITGOOD

/* goodG2B() uses the GoodSource with the BadSink */
static void goodG2B()
{
  char * data;
  char * &dataRef = data;
  char dataBadBuffer[50];
  char dataGoodBuffer[100];
  /* FIX: Set a pointer to a "large" buffer, thus avoiding buffer overflows in the sinks. */
  data = dataGoodBuffer;
  data[0] = '\0'; /* null terminate */
  {
    char * data = dataRef;
    {
      char source[100];
      memset(source, 'C', 100-1); /* fill with 'C's */
      source[100-1] = '\0'; /* null terminate */
      /* POTENTIAL FLAW: Possible buffer overflow if the size of data is less than the length of
source */
      SNPRINTF(data, 100, "%s", source);
      printLine(data);
    }
  }
}

void good()
{
  goodG2B();
}
```

Figure 6. *The solution lies in setting the pointer to a large buffer*

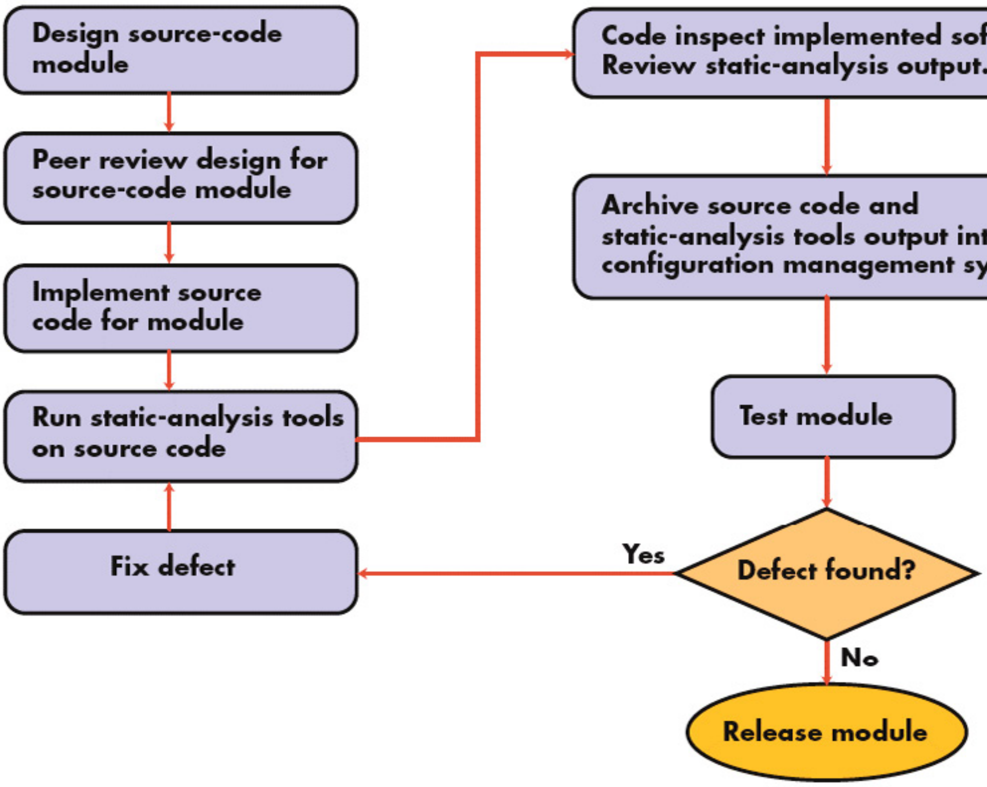

Design source-code module
Peer review design for source-code module
Implement source code for module
Run static-analysis tools on source code
Fix defect
Code inspect implemented software. Review static-analysis output.
Archive source code and static-analysis tools output into configuration management system
Test module
Defect found?
Yes
No
Release module

Figure 7. *This software-development process segment incorporates static analysis [7]*

Risk mitigation

Risk mitigation strategies are four: avoid the risk, reduce the risk, transfer the risk or accept the risk. The overall risk can be avoided by not releasing or developing such software or perhaps using a type safe language in the first place. For the sake of the argument, avoiding, transferring & accepting the risk will not be discussed here. As for risk reduction, it can be done through reducing the vulnerability values, number of vulnerabilities or the likelihood of the security risks to happen. Below is a list of suggested controls that can be implemented to reduce the risk of buffer overflows:

- Using safer compilers
- Disabling the stack execution
- Preventing return addresses from being overwritten
- Reduce the amount of code that runs with root privileges
- Avoiding the use of unsafe functions such as `strcpy()`, `gets()`, etc

References

[1] *http://c85c7a.medialib.glogster.com/media/fa/faeeaf4324a825d97b31f70779d492e6835d5286a1435d178d83db-0106d44ef5/superman-jpg.jpg*
[2] *http://www.embedded.com/design/other/4006735/Integrate-static-analysis-into-a-software-development-process*
[3] *http://www.redhatz.org/page.php?id=22*
[4] *http://samate.nist.gov/Main_Page.html*
[5] *http://cwe.mitre.org/about/faq.html#A.1*
[6] *http://cwe.mitre.org/data/definitions/805.html*
[7] *http://www.embedded.com/design/other/4006735/Integrate-static-analysis-into-a-software-development-process*

Monitoring the effectiveness

Now, it is assumed that everything has been done but how do we know if the controls that we put in place are effective? Well, all what we talked about, so far, was focusing on the static code analysis part. The last thing to do in this cycle is to dynamically check the compiled/running application. In other words, using set of tools to send unexpected parameters and perhaps crafted exploits and check the application response. The goal is to have zero vulnerabilities or flaws. Dynamic vulnerability assessment and fuzzing tools are one way to monitor the security controls in place. Tools such as Nessus, Retina, Nexpose are just a few to mention. Below is a general guideline for software development process: Figure 7.

MOHAMMED ALABBADI

CISSP, Deliver Meticulous Information Security Consultancy & Management Analysis for Decision Support, IS Influencer

Offensive Python

Python was created for fun, but evil hackers use it for profit. Why Python is a new threat for security industry and how tricky Lucifer's kids are – let's talk about it.

According to Wikipedia: "*Python is a widely used general-purpose, high-level programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than would be possible in languages such as C*".

The first statement would surprise a Windows user (how many victims have Python preinstalled?), but MacBooks and Linux servers is a different story. Python supplies by default and it's required by many programs, so uninstalling Python is not an option.

Python is a pro-choice for hackers targeting Mac OS X and Linux, because it's cross platform, unlike binary files it does not require permission for execution and it's absolute not readable from antivirus perspective, so the second statement in Wikipedia is wrong too.

MAxOSX/Flasfa.A

Is a good example. A simple Trojan, not even obfuscated neither encrypted. Only 16 from 42 anti-viruses detect it (link to Virus Total http://goo.gl/gOCXCh). Why I'm not surprised? Because, it's very hard to detect Python scripts by signatures. Scripts are different from binary files, generated by a compiler. The same logic (says, a = b + c) could be represented by infinitive (almost) numbers of ways. The variables maybe stored in different registers, different local variables and these variables could be addresses via different base registers. Shortly speaking, the binary representation of "a = b + c" is not the best signature, but it will work, generating relative low numbers of false positives.

Ah, don't mention these nasty false positives. An anchor is in the ass! Software vendors are upset and pissed off, because if at least one antivirus triggers on a file – an average user will not take a risk to install it. Vendors complain and sometimes it comes to a court case, because the vendor loses money. Nobody won the case (as far as I know), but antivirus company loses money too, especially, if their antivirus becomes too annoying and users chooses the antivirus that whispers "All Quiet in Baghdad".

In my experience TOP 10 anti-viruses detect less that 30% of malicious files at the moment of the first wave of infection. The detection rate is slowly growing up in the next 10 days. After 10 days the given antivirus either detects the disease, either not (because of limitation of the engine, or because the company has no sample).

What else do you expect, dude? You do need a sample to write a signature for it. Period. Somebody somehow has to realize that he or she is infected, find the malicious file and send it to his or her favorite antivirus company. It takes time. Yeah, I know about heuristic and emulation, but... unfortunately nobody created an emulator for Python yet. Why? The answer is simple. Relative low number of Python Trojans makes no business value for it, but requires a lot of money and human resources.

Welcome to the real world, dude. Forget marketing bullshit. Antivirus companies focus on detection the biggest problems to prevent outbreaks. Generally speaking, an antivirus does not prevent infection. An antivirus stops massive diseases. To fight with Python antivirus companies have to write thousands lines of code, create collections of good scripts to check for false positives. Like they have nothing to do. However, sooner or later it will be done and then…

Deeper in the dark water

Python is not always a script. Sometimes is something different. Take the *Cython* project for example (*http://cython.org/*). Cython is an optimising static compiler for both the Python programming language and the extended Cython programming language. The Cython language is a superset of the Python language that additionally supports calling C functions and declaring C types on variables and class attributes. This allows the compiler to generate very efficient C code from Cython code. The C code is generated once and then compile with all major C/C++ compilers in CPython 2.4 and later, including Python 3.x.

Simply speaking Cython is a front-end compiler. It converts Python to C and then the back-end C compiler (any ANSI C compiler such as gcc) compiles C to binary. Why would a hacker do it? As we found out, a binary file is easy to be detected with signatures, so hackers are going to lose the game.

Well, it would be true, but… Cython is Lucifer's brother. Does somebody have poison? Anybody?! Kill me, kill me, but don't ask me to analyze that ocean of dirty water. Cython generates a zillion lines of code in a second. A small script becomes a huge fat program. It's almost impossible to analyze it.

Speaking of signatures – that's the last hacker's concern. Imagine a hacker's server. When a victim sends GET request – the server calls Cython to compile the malicious script to C and then calls gcc with random command like keys, using different optimization techniques, so the code will be different every time.

The generated binary code is too fat and too complex to be analyzed with emulators, and it's too unpredictable to be detected with signatures. A good signature writer can find unique byte sequences, but there is always a risk that these bytes are part of a common library. Says, a hacker found 3rd party cryptography library and used it for profit. Hello, false positives!

Better don't take a job than take it and do it wrong. After all, Cython-based Trojans are minority of minority. Nobody will blame you, if you don't detect it, but for false positives you will be crucified. Even worse. A typical signature writer does not have enough time to solve all cases, so he or she starts with easiest ones and complex cases usually are left unsolved forever.

Coffee break

Java is the most vulnerable platform and target number one for hackers. The classic hit: download-n-execute. Then HTTP request has "Java" agent field and the HTTP response is executable file or a python script – we're under attack. A simple firewall rules can block up 90% of the attacks. How to bypass it?

Grab your mug cup to make some mocha. *Jython* (*http://www.jython.org/*) is an implementation of Python which is designed to run on the Java Platform. It consists of a compiler to compile Python source code down to Java bytecodes which can run directly on a JVM, a set of support libraries which are used by the compiled Java bytecodes, and extra support to make it trivial to use Java packages from within Jython.

It's a good idea, but bad implementation. Java decompilers generate endless spaghetti, giving you a headache and suicidal thoughts. Time comes and goes and you are trying to unravel the tangle. The day is gone. Finally you realized that the decompiled code is wrong and something is missed. To confirm this theory you use Java disassemblers.

Wow! The cycle that changed the variable which was never initialized before and never used after – it's not hacker's bug. It's the decompiler bug. Jython code is too messy and it's different from the native Java compiler. Using Java disassemblers would be an option, but it's time to go home and say "good bye".

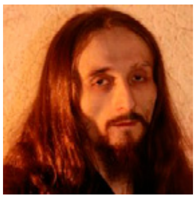
Even if you are brave enough to reconstruct the logic – it does not help you to write a good signature, because you do need special experience to distinguish library code from hacker's code.

On the ocean floor

Lunch time. I mean is time to launch a new torpedo. We talked about different Python compilers, but Python is a compiler too. Well, kind of. When it loads a library first time, Python translates it to byte-code (usually, it has an extension "pyc"). When it's done – you can remove the original Python script and with some limitations run the byte-code on victim's machine.

Now what? You have a byte-code, but unlike Java byte-code, there is no specification for Python neither disassemblers/decompilers. This game has no name. Python is one of the most offensive languages on the planet. Long life to Python!

KRIS KASPERSKY



Is a reverse engineering expert at the top of his field of endeavor. He possesses extraordinary ability and is internationally recognized as one of the top specialists in the field of Reverse Engineering. His exceptional research, rare analytical skills, and extraordinary reverse engineering experience have enabled him to excel and succeed while gaining international acclaim among top industry leaders in the world.

Having fun with antennas and why you need to make your own

Antennas (antennae for the serious people and entomologists) are the most omnipresent and the most misunderstood pieces of tech we all have, and yet, as everything keeps getting smaller and smaller they remain one of the few hacker friendly items we can tinker with. In these few pages, let's have a first basic approach on how they work, learn what's cool about them and get ready to build our own!

Let's play a little game: take a quick walk around your place and try counting how many antennas you have. You should count at least 20 of them: the big one for your TV, your smartphones (GPS/Bluetooth/WiFi/GSM/NFC/Inductive charging), laptops & tablets, your internet modem/router, DECT phones, your garage door (and GDO), wireless keyboard/mouse, your gaming controllers, anything RFID you might have etc …

But do you know how they work? Do you know what an antenna is? Before writing this I went and asked people I know from a non scientific background and almost all of the answers included a „chunk of metal" bit; however saddening it might have been for me to come to the realization that I'm specializing in chunk of metals, it is true that in its most basic form, an antenna is nothing more than a wire.

If like me some years ago you end up wandering the internet looking for information about antennas you will unmistakably find yourself on a ham radio amateur webpage and I can tell you that a lot of them use the term *amateur* because they are humble people: in reality they're more like engineers on steroïds who create designs and experiment like crazy for the sake of being able to speak with other amateurs all around the world. Science bringing people together is always a beautiful thing, if you wish to delve deeper into this world, see how technical and scientific it can get, use words such as *capacitive reactance* in your everyday life, I strongly recommend you to visit some of these blogs since we won't get that much into the details in this article.

Antenna Crash course

Any resource you'll find talking about this topic will teach you about electromagnetism, before even using the work antenna, for today, we'll try to imagine how it all works, without getting too much into the physics.

Fun to imagine

If you take a wire and plug it to a DC generator, the wire will emit a small electromagnetic field, if this wire is coiled, the field will be stronger, if it is coiled around

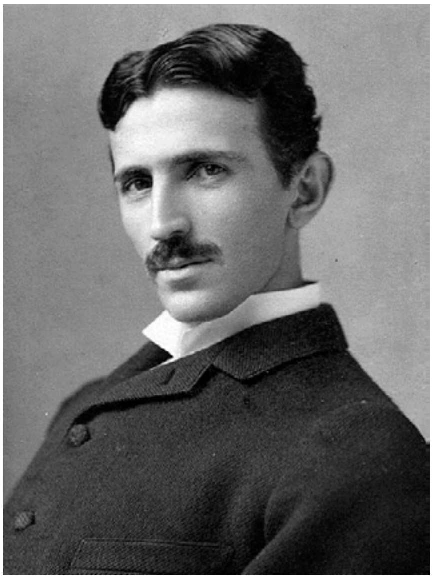


Figure 1. *Nikola Tesla, famous for his discoveries in the AC field*

an iron core, it will become an electromagnet. What's important is that Faraday discovered that if a current goes through a coil and that there is another coil nearby, if we increase the current of the first coil, it will induce a change in the second one across the air.

This electromagnetism propagates through the air and loses power as it travels, reflect, refracts, scatters and diffracts and that's not even counting the potential interferences.

The problem when using DC is that to continually induce, we'd have to continually increase the current which would be problematic at some point and this is why we need to use AC, while in DC the electrons move steadily in the same direction, they do this weird dance in AC where they do two steps forward and then two step backwards, two steps forward again etc … imagine yourself on a rowing machine, every time you pull you create a peak of energy over time but while you're coming back and preparing to pull again this energy diminishes, the number of times you pull each second would be the frequency and if there were a receiving coil next to the rower, the changes happening in the rower would be induced in the coil and the signal received would look like a sine function: this is our carrier wave, the magic can finally happen.

When we divide the speed of light by the frequency, we get the wavelength, that's the famous: λ = c/f formula with λ the wavelength in meters, c the speed of light in m/s and f the frequency in Hz, to be precise, we'd use 95% of c which is the speed of the electricity in a wire.

Let's get back to our antennas, earlier we said we needed a coil on the receiving end, that's not entirely true, let's say we don't care much about receiving the current at the same voltage but we'd very much like to get the information of the signal such as its frequency or the information it carries, to do that we don't really need the whole package, so we don't really need a coil: using a bit of straight wire will allow to receive this information, although it will be much, much weaker than if we had put a coil right next to the emitter: it doesn't matter, we just need to amplify this weak signal again in our receiver, this bit of straight wire is our antenna.

Why size matters

Let's consider the following wave: Figure 2.

What we see here is the full wave length, an entire period/cycle.

We'd like to get that signal into our antenna but ideally, it would be good if the antenna were just long enough so that we'd have a polarity change at the very end of it, when that happens, the antenna is said to be resonant.

However, it is not always possible to have an antenna the size of the wavelength because obviously, at low frequencies, the wavelength is very long, if we were to use a half-wavelength antenna, it would still be resonant since in this example, we have a polarity change right in the middle, it would also work at ¼ and ¾ since the voltage is shifting, for all intents and purposes any antenna which length is ¼ wavelength or a multiple of a ¼ wavelength is resonant.

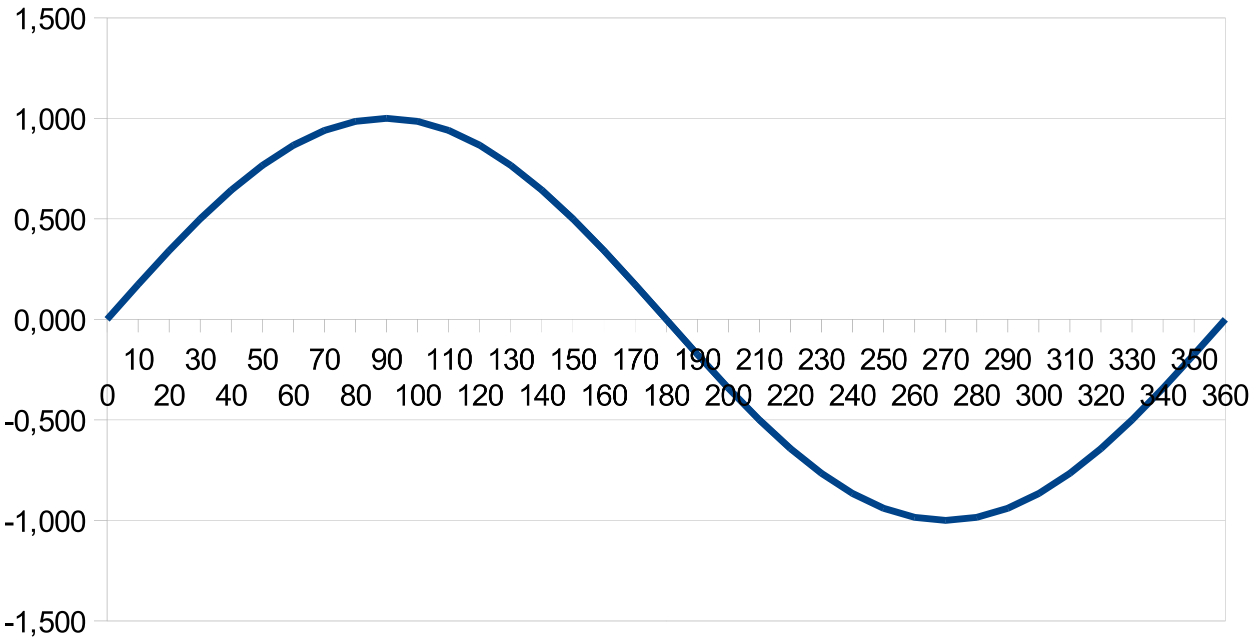


Figure 2. *A simple Sin(0-2π) graph*

If you have a WiFi access point with antennas, chances are they will be half wavelength (around 6,25cm long).

Antenna impedance

A very short word on antenna impedance: the impedance is the amount of resistance the antenna is going to present to the current it will receive, it relates the current and the voltage that goes through it so it's a very important value, when you buy an antenna on the market, the impedance will have been matched to 50 ohms by the manufacturer, this is why you can't just add some length to any antenna, it's unlikely that it will work because extending to your antenna by soldering more metal (even if you keen using ¼ λ multiples) is going to add some impedance to it and unless you match it again you are going to de-tune it.

Radiation pattern

There are many types of antennas and as much radiation patterns, let's have a look at the most important ones:

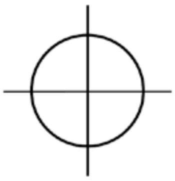


Figure 3. *Isotropic antenna radiation pattern 2D*

The Isotropic antenna is an idealistic lossless antenna that radiates with the same power in every direction, in Figure 3 you can see it as a circle, but we have to keep in mind that we live in a three dimensions world, so the isotropic antenna radiation pattern really looks like a perfect sphere. These antennas don't really exist as such, but we could consider celestial radiators like stars as isotropic emitters.

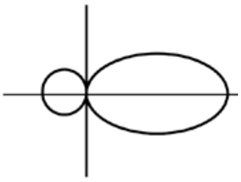


Figure 4. *Yagi antenna radiation pattern 2D*

The Yagi-Uda antenna is a very popular design and you probably use one for your TV, it uses several elements (driven, reflector, director) and is the most widely used directional antenna as you can see on Figure 4, the pattern is much *longer* and narrow. In fact, you simply sacrifice some omnidirectionality so that your antenna will *reach further* in one direction, in the pattern we can see that the antenna emits on the right but also a little on the left, there are two lobes, the main one being the directional one on the right.

An omnidirectional antenna radiation pattern will very much look like the isotropic one in two dimensions, but in 3D, it will look more like a donut than a sphere.

Antenna Gain

We've all seen expensive so-called High-gain antennas on the market but what does it really mean?

The gain is the ratio between the power emitted by the antenna in its main lobe and what an isotropic antenna would radiate in that same direction, the gain is usually measured in units called decibels-isotropic (dBi). A little word of warning here, the decibel is a logarithmic unit, which means that when a manufacturer displays a gain of 3 dB, they are pretending to double the range of your antenna. The problem is, sometime they display much more than that and when you start seeing +9dBi antennas (8 times the range) with the same power level, without any amplification mechanism on the antenna, you can start smiling. This gain measurement is criticized, a lot of people think it is not realistic to compare an antenna to an isotropic one since it can't possibly exist: to have a more realistic approach, the gain can be measures in dBd which is the ratio between the power emitted in the antenna main lobe and what a dipole antenna would radiate in that same direction.

What's important is that the gain is the amount of omnidirectionality you sacrifice to gain directionality at the same emitter power level, the way of achieving this is by having longer antennas, remember earlier when talking about wavelengths, well having a longer antenna is going to achieve a higher gain: there will obviously be no impact for the isotropic antenna since its gain is always 1, a higher gain Yagi-Uda will have a narrower main lobe but which will reach further, the omnidirectional antenna will look a little more like a disc and less like a donut, in shorter terms, your antenna will reach further but you will have to aim better towards the destination.

This is it, we're done with antenna theory, we know all we need to know to be able to understand how they work in a superficial but sufficient way for now.

Some other cool uses of antennas

Antennas have a myriad of applications, but they are very important in some of those, let's have a look at some examples:

Radiotelescopes

It might be a little bit counter intuitive to point an antenna towards the sky and expecting to „see" something, however, by analyzing the values received, we can know what's out there

Figure 5. *The Arecibo radiotelescope*

Sonic Weapons

These weapons have been quite popular these last few years because they are non lethal and less expensive than true weapons on the long run.

You probably heard of the Long Range Acoustic Device (LRAD) which is a way of sending sounds across large distances, these are used on boats, the principle is sound (pun intended), they are meant to be so loud that you stop whatever you're doing and try to get far away before your eardrums give up.

Figure 6. *A Humvee mounted Active Denial System*

The Active Denial System is like a science fiction weapon, it projects an energy beam that excites the water molecules from the surface of its target like a microwave oven does and when this target is your skin it can't possibly go well, in practice it causes you to flee because the burning sensation disappears as soon as you get out of the beam so this is still a non lethal weapon.

TeraHertz imaging

This is the infamous tech used by TSA in US Airports that sees through clothes, in practice it uses wavelength at the border of infrared and microwaves 100µm to 1mm, the challenge here is to have sufficiently small antennas.

That's not all there is, because once we overcome the challenge of having ridiculously small antennas, we'll be able to communicate at extremely high frequencies (over 300GHz) and since the antennas will be at the nanometer scale we can just imagine the MIMO arrays we're going to have with a million antennas in our cellphones.

Now, for the fun part, we are now going to build a WiFi directional antenna.

Building a Cantenna

There is no mystery there, a Cantenna is an antenna made of a can or of multiple cans.

There are many advantages to building a directional WiFi antenna, first of all, it's very cheap to build whereas it's very expensive to buy and there's a reason for it, there is usually very little use in having a directional WiFi antenna at home (modems and access points are equipped with omnidirectional antennas), the other reasons are less obvious but for example, you might need to connect to your neighbours' connection for any reasons and would need to have a better reception, another reason you might want to have a directional antenna might be to use while driving to verify that a side of the road secured their WiFi properly (to alert them if they didn't).

The last reason, is because it's fun: you'll enjoy yourself while doing it and you'll feel like learning more about antennas, maybe the next step will be to build a Yagi Uda array (plus, a Cantenna looks cool, like a radar).

Here's a little disclaimer:

- if you're missing something, if your can diameter is a little too small or a little too big it doesn't matter, do it anyway, learn, experiment,
- if you tinker a little bit, you should have a lot of the hardware required excepe the antenna components which shouldn't exceed 10$ if you

don't have all the tools, ask a friend, buy it or find another way, be resourceful,

- when using anything that is either fast or hot or noisy, protect yourself, get protection glasses, some gloves and a mask, please remember that you only have one set of eyes/hands/ears and that it only takes one mistake to lose them, don't be a hero and get some protection,
- You're not building the ultimate antenna, this is a project for learning and having fun with tin cans!, don't be hard on yourself, if you have fun you'll be building another, better antenna in no time.

You'll obviously need a can to do this, go our and buy one, diameter should be around 8.25cm and as long as you can find: where I live there are a lot of cans smaller than this and a lot of cans bigger, this is a rarer middle type, the one I found was for sliced pineapple.

What we'll use the can for is called a waveguide, it's not the antenna per se, it's a device that allows the waves to travel in a predetermined fashion, in short terms it guides the to the „real" antenna (hence the name).

Figure 7. *A can of sliced pineapple: 8.25cm diameter and as long as I could find*

I couldn't find a longer one than this so I decided to buy two of them, solder them together to have a longer waveguide (we went over this, the longer it is, the more gain it will have, it is also true for a waveguide).

Eat these delicious pineapple slices (or whatever you bought), remove the label, clean the can.

You'll need a 50 ohms coaxial female connector here, the choice of the connector is entirely up to you, you can use N-type, BNC etc ...

Call me old fashioned but I like BNC connectors, they are small, easy to use and they just feel right, it's also easier to switch your cable from one antenna to the other this way.

To mount this connector you'll need to drill through your can at a precise distance from its bottom, this distance depends on the waveguide diameter so if like me you found a 8.25cm diameter can, this point will be at 6.35cm, if the diameter is different, you should use the simple and efficient calculator on this page: *http://www.turnpoint.net/wireless/cantennahowto.html*.

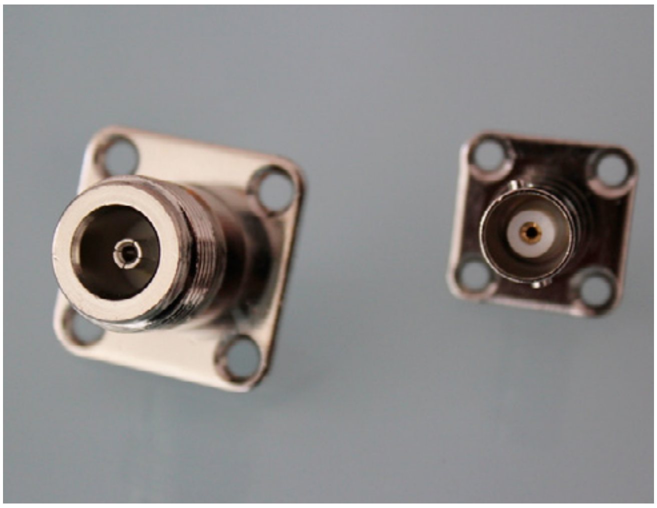


Figure 8. *A N-type female chassis-mount connector on the left and its BNC equivalent on the right*

Use a drill or a nail and a file to make holes big enough for the connector on your can:

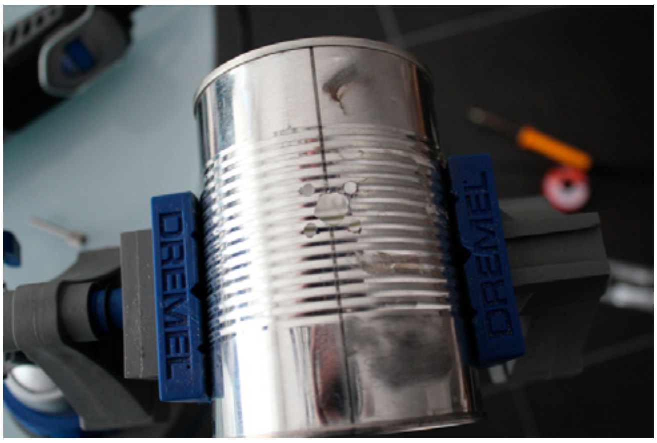


Figure 9. *The drilling pattern for the connector*

Next, solder a piece of 12 gauge / 1.5mm copper to the inside part of your connector, the wire is supposed to be the hard 1 copper brain type, it must be difficult to bend, it must look like this: Figure 10.

Remember earlier when we talked about ¼ wavelength multiples? well the wavelength for the 2.4GHz WiFi connection is 0.125m, here we'll use a ¼ wavelength antenna and cut the line at 3.125cm: try to be as precise as possible.

We only need to mount this to the can, try to find nuts and bolts that fit well and set it: Figure 11.

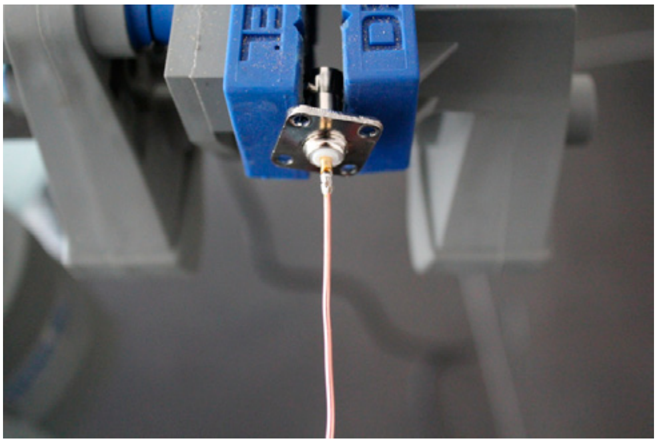


Figure 10. *Our feed line before cutting*

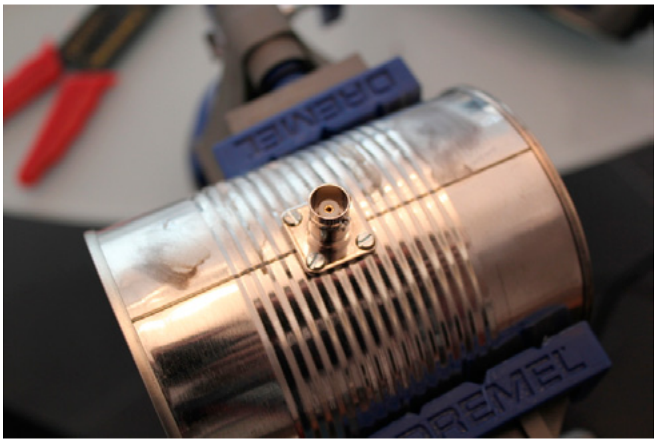


Figure 11. *We're almost done here*

The rest is up to you, mount it on a small stand, which should be cheap, lengthen the waveguide by adding another can, paint it, put stickers on it, name it etc ...

The only last thing you'll need is a pig tail: a cable to connect your antenna to whatever you want to plug it onto, that cable should be a coaxial cable with a male N-Type connector on one end (to plug it to your Access point for example) and the other end should match the female chassis mount connector you used on the can.

Each of these items (cable, connectors) should have a 50 ohms impedance (there are two versions, 50 and 75 ohms), you should also choose the version meant to be soldered, you can also choose not to solder anything and to buy the connectors to be used with a crimping tool but I prefer the solder version because you can unsolder the connectors to use it on other cables.

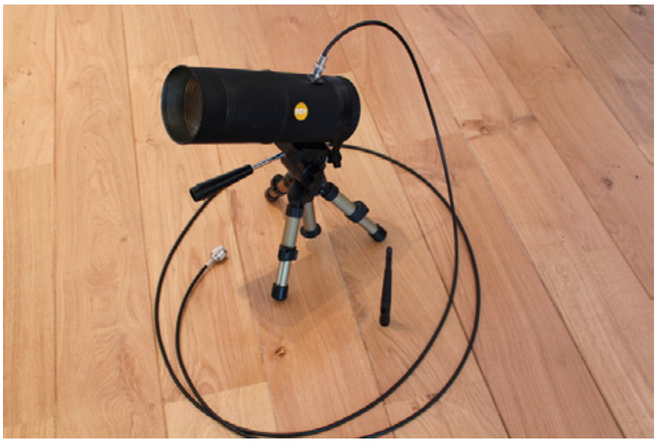


Figure 12. *Our d00mtenna next to a mere mortal omnidirectional antenna*

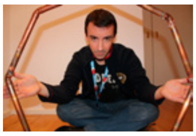
Here is what our creation should look like: Figure 12.

Summary

This is it people, in a short time, we not only learned about antenna theory but we created our own antenna, if you had fun, it might be a good start for you, since you already own all of the pieces, you can experiment with different sizes and try things, just change the can!

It might also be a good introduction to building a Yagi Uda array, in any case, if you wish to learn more about all this, I can't encourage you enough to spend some time on Ham radio amateurs blogs, it was fun sharing all of this with you, I'm usually quite busy but I don't do this nearly as often as I should, thank you for reading.

GUILLAUME PUYO



Guillaume is a French consultant, Security researcher at EURA NOVA in Brussels and a postgraduate student in software and systems security at the University of Oxford. He specializes in wireless security and in all sorts of mad science.

Payment Card Security

There are many standards ensuring minimum level of protection to sensitive information such as the Payment Card Industry Data Security Standard (PCI DSS) which protects the cardholder's data (CHD), Data Protection Act, FSA regulations for financial information and ISO-27001 the information security Management standard. These standards are built to be more generic helping organisations deal with security risks and not to protect them from all security threats. Therefore, this article focuses on PCI DSS and what can be done and what approach must be followed by the experts to ensure security of information not just compliance with PCI DSS.

After thousands of frauds American Express, Visa, MasterCard and other card companies decided to add an additional level of security for card issuers, by forcing merchants and service providers to comply with PCI DSS when they store, process or transmit CHD. As a result, the Payment Card Industry Data Security Standards Council (PCI SCC) was formed on December 2004 and card companies released PCI DSS.

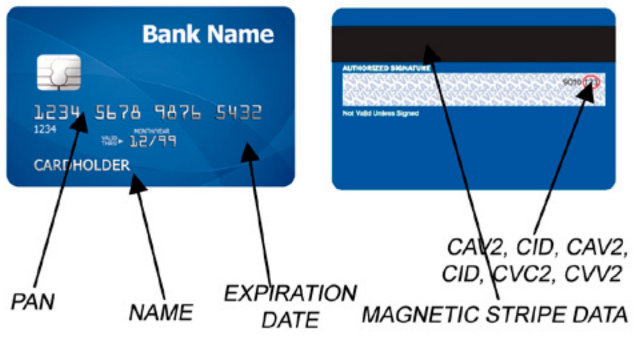


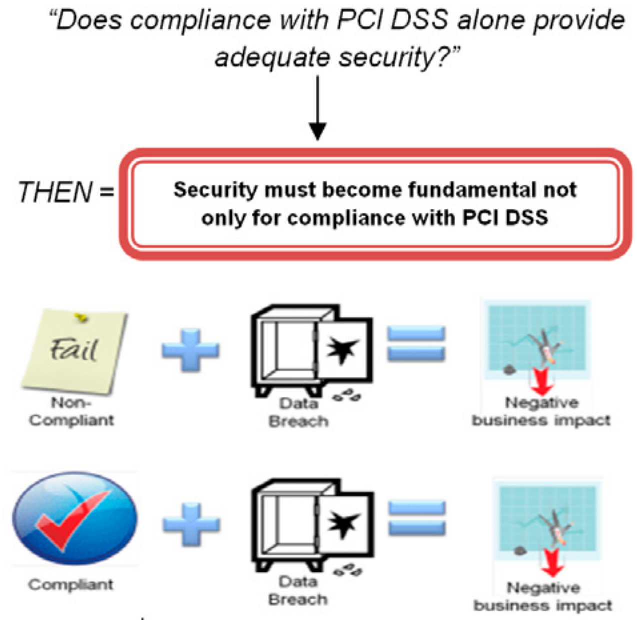
Figure 1. *Payment Card*

PCI DSS purpose is to apply on all entities that transmit process or store CHD. It does provide technical and operational requirements for merchants, acquirers, issuers and service providers as well. Service provider is not a payment brand but it may impact the security of CHD such as service providers that provide IDS, IPS, firewalls. They also store, process or transmit cardholder information on behalf of clients, merchants or other service providers. We consider as CHD, the Primary Account Number (PAN), expiration date, service code and cardholder name. Sensitive Authentication Data is considered as the Card Verification Values (CAV2/CVC2/CVV2/CID), PIN and PIN blocks, Track, Track 1, Track 2 Data and full Track. The difference between general CHD and sensitive CHD is the fact that, sensitive data should never be stored after the authorisation even in an encrypted form.

Basically, card companies set acquiring banks responsible to comply with PCI DSS, and these acquirers ensure compliance with the standard via merchants. At the end, merchants must comply with this standard to protect user's personal data that is being stored, processed and transferred. Eventually, the standard is an agreement between payment card companies, merchant's banks and the merchants. According to the standard, organisations must adhere to twelve PCI requirements and six controls, which are shown in the table below. Therefore, PCI DSS consultancy is required in order to understand the processes, procedures and IT technologies that are needed to the business to achieve compliance with it. However, as explained in the next paragraphs most of the times, compliance does not guarantee security of information within an organisation.

Table 1. *Payment Card Protection*

1. Build and Maintain a Secure Network	R1: Install and maintain a firewall configuration to protect cardholder data R2: Do not use vendor-supplied defaults for system passwords and other security parameters
2. Protect Cardholder Data	R3: Protect stored cardholder data R4: Encrypt transmission of cardholder data across open, public networks
3. Maintain a Vulnerability Management Program	R5: Use and regularly update anti-virus software R6: Develop and maintain secure systems and applications
4. Implement Strong Access Control Measures	R7: Restrict access to cardholder data by business need-to-know R8: Assign a unique ID to each person with computer access R9: Restrict physical access to cardholder data
5. Regularly Monitor and Test Networks	R10: Track and monitor all access to network resources and cardholder data R11: Regularly test security systems and processes
6. Maintain an Information Security Policy	R12: Maintain a policy that addresses information security



"Does compliance with PCI DSS alone provide adequate security?"

THEN = Security must become fundamental not only for compliance with PCI DSS

Figure 2. *What happens to the information*

Information Security and PCI DSS

PCI DSS is only responsible to protect CHD and nothing else. However, organisations store more than CHD which must remain secure and are being kept to paper, hard copy, databases, spreadsheets and IT systems depending on what the business does. Some of that data is confidential and must remain as it is, such as health records, client's passwords, government sensitive information, client's videos and pictures, therefore Risk Management (RM) makes its appearance. RM then deals with all kinds of information and not only with CHD, because PCI DSS compliance leaves rest of the data remains with minimum protection against potential threats. The million dollar question then is:

> *"What happens to the information other than CHD?"*

Other than CHD?

The best way to manage your business information (other than CHD) then, is by employing ongoing *Risk Management* programme, which includes the processes and coordinated activities to direct the whole organisation. Now, Risk Assessment as part of RM is required for PCI DSS too to identify the threats and vulnerabilities of critical functions, assets and components. RM also provides assurance that your organisation keeps controls cost-effective and proportionate to the risks. Firms need both compliance with PCI DSS to protect CHD and RM to secure and manage risks around the business from security breaches on all critical information.

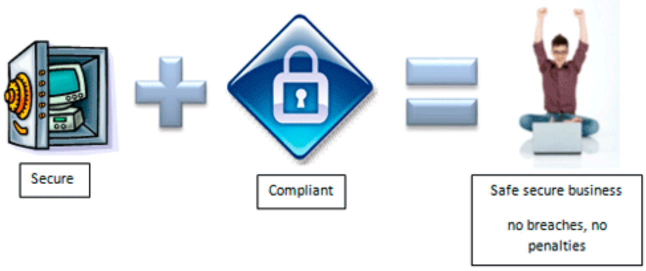


Figure 3. *Risk Management*

RM provides alignment between business and information security that suits the culture and requirements of your business involving the stakeholders to take critical and cost decisions. Sometimes organisations prefer to implement only security practises which are not enough to avoid RM and provide 100% security. This does happen since most firms are dealing with technology assets and operations on daily basis, hence a true RM program must be followed on all sectors. As shown in Figure 1 there are three steps that must be followed to provide compliance with PCI DSS. However, if we think RM is needed to manage general information then it is based on the following six activities (Figure2):

- Asset Identification
- Business Impact Assessment
- Control Assessment – Gap Analysis
- Risk assessment
- Risk Treatment
- Implement agreed Risk Treatment controls and measurements

Risk Management process

- Identify the assets of the organisation, understand their importance and their type, their location and define the owner. Categorise the assets as relevant to the organisation's business, by structuring them to different groups regarding their functionality and purpose. For example the information asset list contains:
  - A name and description of the asset.
  - The owner of the asset / asset group.
  - The IT systems the information is stored/processed on.
- The next step includes Business Impact Assessment which depends on the Availability, Confidentiality and Integrity (CIA) of business's assets. In order to calculate the asset value, the owners need to identify the consequences if a security breach occurs affecting the CIA. For example the table below presents the different levels of assets and the consequences when they are breached regarding financial, legal and repudiation damages (Table 2).
- Perform Gap Analysis or Control Assessment to identify gaps that may exist and improve them. It is a way to compare current controls and practises helping you find any gaps and areas that suffer from threats and mitigate the security risks. Decide whether the implemented controls are acceptable to mitigate the risk and evaluate the risk.
- Employ Risk Assessment which calculates the risk value to estimate its significance.

  What the risk is?

  Risk is the potential that a given threat will exploit vulnerabilities of an asset and hence cause harm to the organisation (financial, Legal or Damage impact). As a generic process you walk around with the employees, interviewing them and look what could reasonably cause harm to find any weaknesses and eventually evaluate the risk. Since the information has been gathered by the control assessment, interviews with colleagues and interesting parties Risk assessment is responsible to identify risks and vulnerabilities. The risk value is calculated by multiplying the impact value of the asset, by the likelihood of a risk to happen by the thread level. Likelihood it's referring to the possibility a threat to exploit vulnerabilities, something happening.

Table 3. *Likelihood/Possibility*

Likelihood / Possibility	↕	It is possible that there will be a security breach within the next three years
		It is unlikely that there will be a security breach within the next three years.
		A security breach within the next three years will not occur.

- Last but not least is the planning and implementation of the risk treatment process. This process depends on the risk value and is taken place once the risk has already been identified and measured properly. If the risk falls within the fault-tolerance the team decided to *accept* the risk. When the impact is too high and the thread happens frequently, then the business must simply do not implement the specific actions and *avoid* the risk. The selected team also can *transfer/share* the risk in order to reduce the burden of loss when the events were to occur. The major consideration thus must be taken when you need to *reduce* security risks, hence the appropriate level of Management then needs to approve appropriate countermeasures. As a result, risk treatment lead us to determine the appropriate controls for reducing the risk, the impact of potential threats and the likelihood of a threat to take advantage a vulnerable asset.

Table 2. *Different levels of assets and the consequences when they are breached regarding financial, legal and repudiation damages*

Asset Value	Impact	Financial Impact	Legal Impact	Repudiation Damage
1	No Impact	No Financial Impact	None	None
2	Minor Impact	Minor financial impact	None	Staff aware, loss of morale, single customer aware
3	Some Impact	Some financial loss	Breach of laws, regulations or contract leading to litigation or prosecution & fines	Multiple customers and businesses aware, local media coverage
4	Serious Impact	Significant financial loss	Breach of laws, regulations or contract leading to litigation or prosecution & significant fines	Widespread local or limited national media coverage
5	Business Threatening	Major financial loss / business threatening	Breach of laws or regulations leading to prosecution & possible imprisonment	Widespread national media coverage

Following the previous steps at least annually gives a clear vision to the management team how the business is coordinated having information security in mind. It also keeps them up-to-date if changes happen in critical operations and services and how to control any vulnerabilities related to PCI DSS and other information. Risk Assessment benefits organisations meets the requirements of PCI DSS and find additional controls to reduce risks and not to bypass them.

RM in alignment with PCI DSS requirements is a guide to organisations on how to effectively apply the above principles in order to manage security risks identified by not increasing security risk. It supports the business process and helps to engender and maintain costumer trust in a business process or service by ensuring that the costumer receives a consistent service and the quality of the service is preserved. Essential for the business is the fact that RM must be continuous in regular intervals to help organisations deal and mitigate significant threats, vulnerabilities and risks in effective manner.

Therefore, as highlighted to the previous paragraphs obviously organisations must be fully aware about the information they are dealing with and be able to protect all of them. PCI DSS only protects CHD and all other data are exposed to critical threats. As a result we introduce the term of RM, its steps, its critical aspects and benefits.

Critical Aspects of Risk Assessment

- Asset identification and classification: Information must be gathered from all stakeholders and grouped properly. Human resource, IT, business and financial department's staff must go through number of interviews with the expert in order to determine the processing, stored and transmitted data such as PAN, expiration date, and service code and cardholder name.
- Risk Assessment must stay simple as much as possible and built in structure methodology. The methodology followed by the experts must be developed and implemented according to the firm's needs to evaluate the risks. The prior goal is to protect CHD by reducing the risks, using appropriate controls that will be validated by the individuals who perform the risk Assessment.
- All employees must be part of the training and awareness programme, in order to understand the importance of information security and how is related with PCI DSS. In addition to that, they must be aware of the impact of a security breach and how to deal with it when a threat exploits a weakness on the CHD.

Benefits of Risk Management

- Alignment and integration between information security and business.
- Manage risks related on all information.
- Areas with sensitive information can be identified for further investigation to find potential threats.
- Raising assurance in the security of information and systems in a business environment.
- Overview of the business-related risk, investing according to the importance and classification of assets.
- Protects the repudiation and public image of the firm.
- Having an effective RM in place shows the commitment of management team to loss reduction and prevention.
- Companies that handle information security on behalf or relating to other companies (providers and consultancy services) benefit, since the above mentioned commitment attracts new customers.
- Management is involved in information security and have always access to information.
- Select appropriate, adequate and proportionate controls to protect information assets and give confidence to third parties.

Conclusion

The reason for this article is to present PCI DSS and the requirements that organisations must satisfy in order to protect CHD from security breaches. However, there exist information that differs from CHD and organisations must also consider alternative solutions to provide and manage security related to those kinds of data. Therefore, we demonstrate the ongoing process namely Risk Management which must be followed by organisations to provide an additional level of security to their assets regarding CIA. This process helps organisations to understand the impact when assets are corrupted and estimate the risk value depending on the threat's level, the likelihood and impact. By evaluating the risks organisations are also able to address security issues in effective manner and put in place appropriate controls and measurements to secure critical operations and assets.

MARIOS ANDREOU

Marios Andreou obtained a BSc in Computer Science at University of Crete in 2011 and completed his MSc in Information Security from Royal Holloway in 2012 (The University of London's Information Security Group). He is an information security enthusiast and he is interested in the area of IT, Software development, Network and Software security, Cryptography and Security consulting.

Evidence Analysis

Welcome back to the Novice approach to Evidence Analysis!! By putting the title to be one of being a novice, I really mean it to be novice – simple,straight and as it is. There can be no alteration done to the elementary alphabets ABCD ... Agreed?? (btw I know the other 22 alphabets as well ;))

So let's get back to some serious elements of Information Security from where we bid it a goodbye!!!

We get back to the three fundamental arms of Information Security, the CIA triad. Also, to the other two arms, that came as Information Security grew older!!!

So we have these five arms of Information Security:

- Confidentiality
- Integrity
- Availability
- Authenticity
- Non Repudiation

Let us see by an example, as how a measure that guarantees security of the information or data achieves these fundamentals.

Mr. A signs a contract with Mr. B. Mr. A sends the asked details via an e-mail to Mr. B by digitally signing the document and encrypting the mail. (Here I have assumed the encryption to be of public-key type, where the same key is used to encrypt and decrypt the message.)

Confidentiality

While a document is being encrypted, it in turn means, that it can be decrypted only through the possession of the key that is meant to. So the message remains confidential in the route to anyone unintended.

Integrity

Integrity means that the data or the information should not be modified in a manner that can not be detected and is done by any means that is unauthorized. Now it is interesting as a slight change in the document will completely change the whole encrypted message (skipping details for the benefit of your heads and Google!!!). That way it achieves integrity.

Availability

Message remains available to both Mr. A and Mr. B (unless their storage space gets over or the deal gets into legal offices with the messages being shredded).

Authenticity

The message remains authentic as it has been signed by Mr. A's digital signature which is unique to him. Also the key that is used to encrypt the message is unique (however, a better idea is always to use private-key encryption).

Non- Repudiation

It means ... If I killed your senses by making you read this, I will say "Yes" if asked ... LOL!!!

It means that Mr. A can't deny the fact that he sent the message to Mr. B. It is accomplished as the message has been signed by Mr. A using his digital signature and that is unique to him.

In this way, we see how a measure taking care of the security of your information makes a goal at all the 5 goal-posts (will try something other than soccer next time!!!).

P.S.: Technically they sometimes differentiate between the literal meanings of *Data* and *Information.* In real life, they are mostly used something like, if one of these goes on a vacation from your mind, you use the other. As simple as that!!!

Now, with this we end the fundamentals and with the next issue we get on to something that gets your CPU on a run.

Next issue will deal with:

- Data Acquisition: A First Responder's Approach
- The Fundamentals of Digital Cloning
- Keep Reading. Be Safe.

Mail me at write2mudit [at] outlook [dot] com.

MUDIT SETHIA

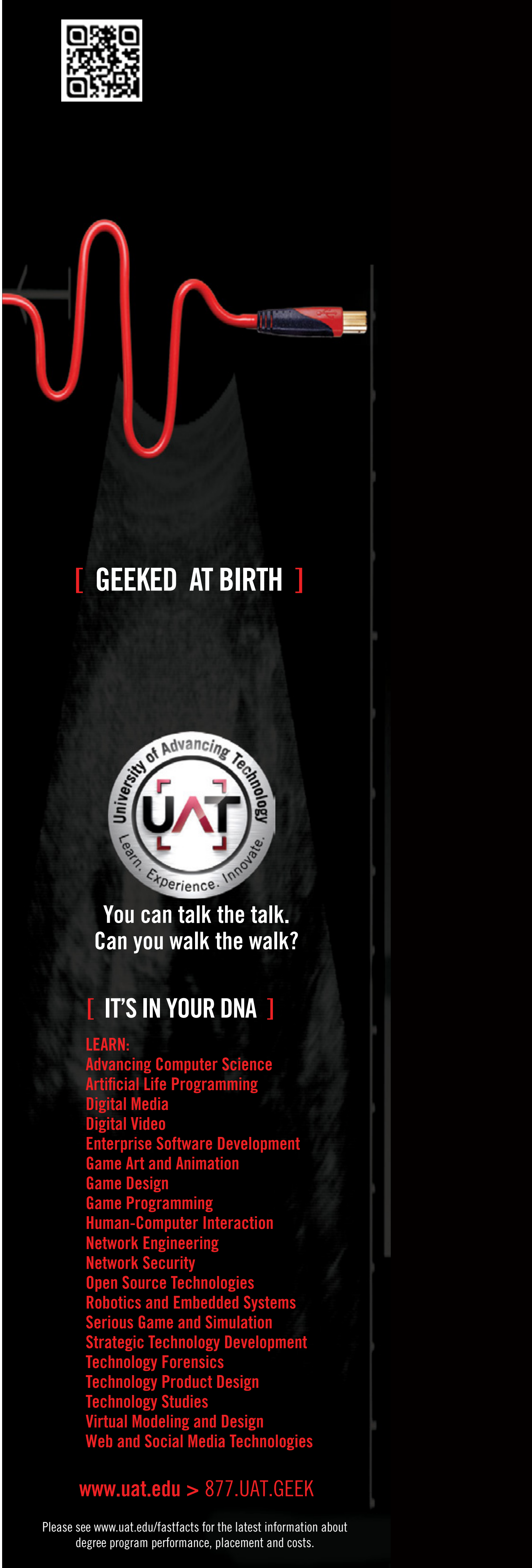
I am a young tech-security enthusiast with special interest in technical as well as the legal aspects of Information Security. Have a certification in Digital Evidence Analysis and Cyber Laws.

I love everything that is related to technology.

Also, I love music,travelling,adventure and CELL PHONES.

Aim: To create a more safe "Webosphere" by creating awareness.

Connect to me at: write2mudit@outlook.com.



HACKTIVITY

Az IT Biztonsági Fesztivál Kelet-Közép-Európában

2013. október 11-12. MOM Kulturális Központ, Budapest

Press Release, September 5, 2013

More information: Bernadette Kis
+36 20 917 9539
bernadette.kis@hacktivity.com

Charlie Miller, Twitter's Senior Security Engineer and Mikko Hypponen, F-Secure's Chief Research Officer at the 10th Hacktivity Conference!

Hacktivity, the largest independent IT Security Festival in Central and Eastern Europe will be held for the 10th time on October 11-12. 1,200 hackers from all over the world, presentations in two simultaneous sections, interactive workshops in small groups, new section for product demos, hardware hacking, lockpicking, live shows, 24-hour hacking competition, a real festival mood and jubilee party in the evening!

This is how you can describe the 10th Hacktivity the full programme of which you can find on the official conference website. Speakers in the programme had to submit their papers for selection (just like at other major IT conferences in the world) as being a speaker at Hacktivity counts as a professional recognition. We received over 50 Hungarian and international papers for the two rounds of the Call for Proposals in three formats: traditional presentation, hello workshop or product demo.

The conference will be opened by **two IT security „stars". The first keynote speaker is Charlie Miller, Twitter's Senior Security Engineer and the other is Mikko Hypponen from Finland.** Charlie worked for NSA for five years, hacked Macbook Air and iPhone among others, and made a fool of the developers of Safari and the Safari mobile app. His Twitter profile says: **'I'm that Apple 0day guy'.** Mikko Hypponen is the Chief Research Officer of F-Secure in Finland. He has been working with computer security for over 20 years and has fought the biggest virus outbreaks in the net, including Loveletter, Conficker and Stuxnet. His TED Talk on computer security has been seen by almost a million people and has been translated to over 35 languages. His columns have been published in the New York Times, Wired, CNN and BBC.

Following the two keynote speakers a total of 32 presentations will be held in the two halls, covering a wide range of fields. László Bíró's presentation on electromagnetic data leakage sounds very exciting just as the story of how a CCTV system was hacked, told by Benjámin Tamási. The presentation of Paul M.

Wright and László Tóth focuses on the security of the latest Oracle database manager (12c – did Oracle finally succeed in designing a secure database manager?). Gábor Pék will give a hands-on presentation on the security issues of hardware virtualization – anyone using it (everyone?!) must attend. The talk on the security issues of SAP deployment servers may surprise corporate professionals. There will be plenty of topics for those who do not think 10 equals two (no typo?!): cyber war, data management in the cloud, act on information security, use of facial recognition tools and many many more subjects. You can read about them in more detail in our newsletters until the date of the conference. Subscribe to our newsletter to stay informed!

Launched two years ago, Hello Workshops became an instant success. This year there will be an even wider selection of diverse topics: in addition to the most popular and perhaps less scientific workshops on XSS, OWASP there will be more in-depth workshops on DEP, ASLR, Exotic injections. Or if you are interested, you can practice Threat Modelling, the Swiss army knife of IT security risk analysis.

There'll be a new round of HACK 24, the 24-hour hacking competition for valuable prizes, a hardware workshop and lockpicking with plenty of new features, numerous games and a brilliant jubilee party in the evening where we'll reminiscence about the beginning.

According to Lajos Antal, Head of IT Security and Data Security at Deloitte Zrt., IT security is one of the most dynamically growing IT, legal and economic fields. As the key objective of the majority of cyber criminals is material gain, not only IT specialists but the senior management of a company must be aware of the risks and jointly do all in their power to set up the controls necessary for an appropriate defence. Hacktivity brings together experts who are outstanding in their fields and the decision-makers, that is why Deloitte decided to be the Diamond Sponsor of the conference for the third time.

At the conference certified (CISA, CISM, CISSP) specialists can collect CPE points necessary for their licence. The organizers expect 1,200-1,500 participants this year.

Further information, detailed programme and registration: *http://www.hacktivity.com*
Facebook: *http://www.facebook.com/hacktivity*
Video of Hacktivity 2012: *http://www.youtube.com/watch?v=mahvRacznho*

A BZ Media Event

Big Data gets real at Big Data TechCon!

Discover how to master Big Data from real-world practitioners – instructors who work in the trenches and can teach you from real-world experience!

Come to Big Data TechCon to learn the best ways to:

- Collect, sort and store massive quantities of structured and unstructured data
- Process real-time data pouring into your organization
- Master Big Data tools and technologies like Hadoop, Map/Reduce, NoSQL databases, and more

Over 60 how-to practical classes and tutorials to choose from!

- Learn HOW TO integrate data-collection technologies with analysis and business-analysis tools to produce the kind of workable information and reports your organization needs
- Understand HOW TO leverage Big Data to help your organization today

"Big Data TechCon is loaded with great networking opportunities and has a good mix of classes with technical depth, as well as overviews. It's a good, technically-focused conference for developers."
—Kim Palko, Principal Product Manager, Red Hat

"Big Data TechCon is great for beginners as well as advanced Big Data practitioners. It's a great conference!"
—Ryan Wood, Software Systems Analyst, Government of Canada

"If you're in or about to get into Big Data, this is the conference to go to."
—Jimmy Chung, Manager, Reports Development, Avectra

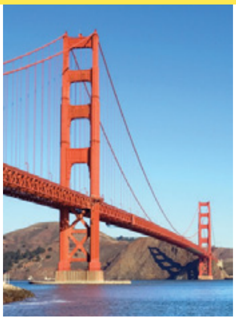
BigData TECHCON

San Francisco

October 15-17, 2013

www.BigDataTechCon.com

The HOW-TO conference for Big Data and IT professionals



Big Data TechCon™ is a trademark of BZ Media LLC.