ECTester

The Unofficial Guide

Dr. Ján Jan#ár

ECTester: The Unofficial Guide

by Dr. Ján Jančár

Publication date 1 August 2019

Table of Contents

Chapter 1. Introduction

ECTester is a Java based elliptic curve cryptography security testing and evaluation tool (ST&E) written by Ján Jančár. The tool tests behavior of elliptic curve cryptography implementations on JavaCards (`TYPE_EC_FP` and `TYPE_EC_F2M`) and selected software libraries.

ECTester also produces heat maps to display the leaks as a dot density graph.

About this Manual

The ECTester book has five chapters. The first chapter details ECTester setup. The chapter discusses how to setup a rig to test Java and native libraries. The second chapter is Java SmartCards. The chapter discusses how to test Java SmartCards. The third chapter is Java libraries. The chapter discusses how to test Java libraries. The fourth chapter is Native libraries. The chapter discusses how to test native libraries like Botan, Crypto++ and OpenSSL. The final chapter is an appendix. The appendix provides CVE findings discovered in Java SmartCards, Java Libraries, and Native Libraries.

Architectures

The examples in the book use i686 and x86_64 for testing. ECTester will work equally well on other architectures, like ARM, Aarch64 and PowerPC.

Chapter 2. ECTester Setup

This chapter discusses system requirements and ECTester setup on Linux. Windows Setup will be similar to Linux.

Requirements

ECTester is a Java based application. Java NNN is required to build and run the program. You should install the Java NNN SDK to install the Java compiler and other SDK tools. In addition you should install `ant` package.

Java SmartCard testing requires XXX.

Java Libraries testing requires XXX.

Native Libraries testing requires common build tools, like make, a C or C++ compiler, an assembler, and a linker. You may also need Autotools like autoconf and automake, depending on the library.

On Debian and derivatives you can install the `build-essentials` package. On Red Hat systems, like CentOS and Fedora, you should install the `gcc`, `gcc-c++`, `make`, `autoconf` and `automake` packages.

Section title

TODO: discuss the details of ECTester setup.

Chapter 3. Java SmartCards

TODO: discuss testing Java SmartCards.

Section title

TODO: discuss the details of testing Java SmartCards.

Chapter 4. Java Libraries

TODO: discuss testing Java libraries.

Section title

TODO: discuss the details of testing Java libraries.

Chapter 5. Native Libraries

Many popular elliptic curve libraries are written in C, C++ and other languages. This chapter discusses how to test native libraries like Botan, Crypto++ and OpenSSL.

There are fours steps to test a native library using ECTester. The first step is build the library. The second step is build `ECTesterStandalone.jar`. The third step is running the collection to collect data. The fourth step is plot the graphs to visualize the data.

The steps below use Crypto++ as an example. You should use the appropriate library name when testing other libraries, such as Botan, Crypto++ or OpenSSL.

Build library shim(s)

Either go to `src/cz/crcs/ectester/standalone/libs/jni` and make that. Or do `ant -f build-standalone.xml libs`.

This will compile the shared libraries which ECTester uses via the Java Native Interface to work with native libraries. That makefile uses `pkg-config` to find Crypto++, so all of that should apply.

Build ECTesterStandalone.jar

Build `ECTesterStandalone.jar` with `ant -f build-standalone.xml jar`, possibly run this twice.

Run the collection

Go to `dist/` and run the collection.

```
java -jar ECTesterStandalone.jar list-libs
```

It should list Crypto++ as available, together with other libraries that ECTester was able to compile the shims for (they were available in the system) and with pure Java libraries. If it doesn't list Crypto++, something went wrong with the above steps.

Then do something like the following.

```
java -jar ECTesterStandalone.jar ecdsa \
  -n 500000 -nc secg/sect233r1 \
  -o out.csv Crypto++
```

That should perform 500k signatures over the sect233r1 curve and output the data into out.csv.

Plot the graphs

Go to `util/`, run the `plot_dsa.ipynb` Jupyter notebook and follow the instructions there to plot the ECDSA data from out.csv.

Appendix A. CVE Findings

The tables below presents Common Vulnerabilities and Exposures (CVE) found by ECTester.

Java SmartCards

The table below shows vulnerabilities discovered in Java SmartCards.

SmartCard	CVE	Comment
Unknown	CVE-2019-NNNN	Unknown

Java Libraries

The table below shows vulnerabilities discovered in Java libraries.

Library	CVE	Comment
BouncyCastle	CVE-2019-NNNN	Unknown
Sun EC	CVE-2019-NNNN	Unknown

Native Libraries

The table below shows vulnerabilities discovered in native libraries.

Library	CVE	Comment
BoringSSL	CVE-2019-NNNN	Unknown
Botan	CVE-2019-NNNN	Unknown
Crypto++	CVE-2019-14318	Information leaks in prime fields and binary curves.
Intel PPC	CVE-2019-NNNN	Unknown
libgcrypt	CVE-2019-NNNN	Unknown
libtomcrypt	CVE-2019-NNNN	Unknown
MatrixSSL	CVE-2019-NNNN	Unknown
MbedTLS	CVE-2019-NNNN	Unknown
Microsoft CNG	CVE-2019-NNNN	Unknown
OpenSSL	CVE-2019-NNNN	Unknown
wolfSSL	CVE-2019-NNNN	Unknown

Index