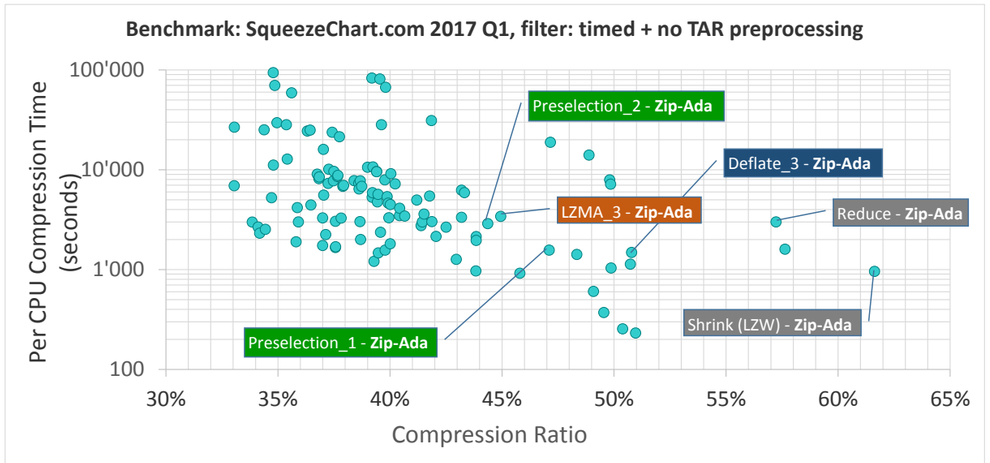
5. Recent developments in the Zip-Ada library

Gautier de Montmollin



Part 1: Overview; a new Deflate compression algorithm

Like many things in software, Zip-Ada started as a hobby project, but over the years has met the needs of several professional users who need the file archiving features or the compression features to reduce storage and shorten transmission times. The presentation will show the evolution of this full-Ada library and the advantages of using it.

The core of the presentation is about recent developments which were needed to make Zip-Ada efficient on the data compression side, while keeping full compatibility with other software reading Zip archives. It will show how we have combined two known and fast open-source compression components for crafting an efficient method for the widespread Deflate format, and developed on top of it a single-pass algorithm which detects "on the fly" changes in the input data flow to further improve the compression. This strategy helps to reduce the compressed data size while keeping the compression time reasonably short.

Part 2: a new LZMA compression algorithm; “Preselection” algorithm-picking

The next step for improving Zip-Ada was to go beyond the classic Deflate. The LZMA format was a candidate of choice since the “LZ” part can be reused from Deflate, and there is already a full decoder with Zip-Ada since 2014. The LZMA format is really amazing on large, structured data, even with simple compression algorithms at work. Typical compression rates on database files or large numerical CSV files are below 10%.

The presentation will show the beauty and the power of the “MA” part of LZMA in a visual way – it is fascinating that the encoder in its simplest version holds in only 300 lines of Ada code, but does a better job than what the most sophisticated Deflate algorithm can do!

It will also present a so-called “preselection” feature for the Zip archiving process, which picks one of many different compression formats and algorithms depending on hints like the stream’s size, or data type.

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

Ada
Switzerland

1st Swiss Ada Event

Thursday 21-Sep-17

HSR Rapperswil

14:00

The event will consist of six half-hour presentations. The first block of three will start at 14:00 and finish around 15:30 for a coffee break. The second block will start at 16:00 and go on to 17:30 after which there will be an Apéro with networking opportunities.

Entrance will be free and is open to anyone, not just members of Ada Switzerland.

However, for logistical reasons, we request that everyone who plans to attend the event register by sending an email to registration@ada-switzerland.ch containing their name, address and contact email.

HSR Rapperswil is only a few minutes' walk from Rapperswill SBB railway station and is clearly signposted.

Presentations: (not necessarily in order that they will be given).

1. Astronomical Ada

Ahlan Marriott & Urs Maurer, White Elephant GmbH

SkyTrack is a PC program written entirely in Ada that is used to control astronomical telescopes.

The presentation is an experience report on the problems we had and the solutions we subsequently developed when we attempted to make SkyTrack host platform independent.

Our goal was for SkyTrack to be available under Windows (XP and later), Linux (Ubuntu) and OSX (Sierra).

The biggest problem we had concerned our decision to implement the program's Graphical User Interface using Gtk. Although AdaCore provide bindings to Gtk in their GtkAda product, this, by itself, is insufficient.

In order to use Gtk a lot more work has to be done and a large part of the presentation is devoted to how we solved this problem.

2. A common time base for parallel jobs in a distributed and embedded real-time system.

Peter Vogelsanger, Rheinmetall Air Defence AG

The presentation describes the common time base of a fire control system for guns and missiles. A fire control system measures an air vehicle and calculates a track of that air vehicle for the lead calculation of the gun or the designation of the missiles. The fire control system has three nodes. One node is the graphical user interface computer which manages the user interaction with the two other nodes. The two other nodes are embedded real-time computer nodes. One of them controls all sensors, like radar, infrared and TV camera, laser for distance measuring and the servo motors. The last node is the central communication and lead computation node which controls the different connected weapons and the communication to a higher echelon.

A common time base is evidently important for a measurement of the position of the air vehicle. The system can calculate the velocity and acceleration of the air vehicle from the position with the time base. There are two possible algorithms for the calculation of the velocity and acceleration of an air vehicle. One algorithm measures the positions and the exact time of measurement (time-stamped). The time-stamp mechanism is almost fully a software solution. The other way is to use a clock-base cyclic measurement system (like a clocked CPU). The clock-based mechanism is a mixture of hardware and software. Both mechanisms have their benefits. We chose both, clock-based for internal time-base and time-stamped when data leaves our system.

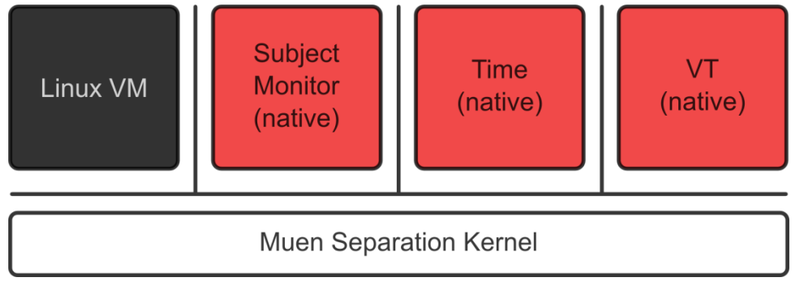
The presentation shows how Ada helped us to implement the timing. It will also show that Ada could help with tasking, object-oriented programming, and accessing low-level features.

3. Muen – An x86/64 Separation Kernel for High Assurance

Reto Bürki & Adrian-Ken Rüegsegger, HSR Rapperswil

Writing large error-free software is extremely challenging or even infeasible. In order to be able to assure critical security properties it is therefore necessary to decompose the system into small security critical subjects whose correctness has to be shown and other large uncritical parts which cannot endanger security.

A separation kernel can be used to assure the independent execution of multiple subjects and the enforcement of predefined communication channels between subjects. The correctness of the separation kernel is therefore essential for overall security.



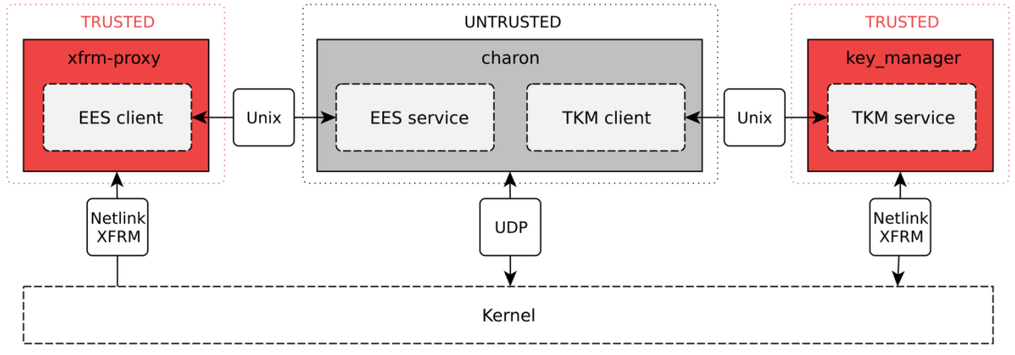
In this presentation we describe the design and implementation of the Muen Separation Kernel which uses the SPARK language to enable light-weight formal methods for assurance. Besides a discussion of software complexity vs. simplicity, system integration, as well as present and planned verification we demonstrate how Muen enables the construction of high security systems on x86 hardware.

4. TKM – An IKEv2 key manager written in Ada 2012

Reto Bürki & Adrian-Ken Rüegsegger, HSR Rapperswil

The IPsec protocol relies on the correct operation of the Internet Key Exchange (IKE) to meet its security goals. The implementation of the IKE protocol is a non-trivial task and results in a large and complex code base. This makes it hard to gain a high degree of confidence in the correctness of the code.

This presentation outlines a practical approach on how to split monolithic security-sensitive software into trusted and untrusted components and implement it as a component-based system to attain a higher level of assurance. By formulating desired security properties and identifying the critical components of the IKE protocol, the key management system is disaggregated into untrusted and trusted parts.



We will illustrate the methodology and best practices by examining the Trusted Key Manager (TKM) which is written in Ada. A specific example how the language choice helped to increase the confidence of operation according to the specification is Design-by-Contract aspects introduced by the Ada 2012 language revision.