Avraham “Abe” Bernstein | CV

S/W Engineer, Architect, Cybersecurity, Algorithms, Innovator

Represented by Tk Open Systems Ltd.

Version: 4.0.4-tkos-abbrev

Last update: 2017-10-11

0.1 Contact Info & Links

email: Avraham.Bernstein+cv+tkos@gmail.com

geolocation: Jerusalem Israel, **tz:** UTC +02:00/+03:00 [winter/summer]

tel-IL-mobile/whatsapp: +972.54.641-0955

tel-US-mobile: +1.845.402-0023

www-home: http://purl.org/Avraham.Bernstein

linkedin: https://www.linkedin.com/in/AvrahamBernstein

cv-abbrev: HTML (easiest to read), PDF, DOCX **[this version]**

cv-full: HTML (easiest to read), PDF, DOCX

1.0 Summary

I am an expert in S/W engineering, cybersecurity, algorithms, and domain specific languages (DSL). I am an experienced “hands-on” computer scientist and S/W architect for industrial strength apps. I devised innovative algorithms and solutions for many S/W problems in a wide range of fields, including:

- **cybersecurity & cryptography:** Viaccess-Orca, Cisco-NDS, Anonymous
- **data science, big data, & database:** Viaccess-Orca, DEC & Iscar, Michlala College
- **bioinformatics:** Syntezza
- **transportation vehicle route guidance:** Telequest
- **blind vision:** Virtouch
- **accessibility & biofeedback:** Cubital, Self-Startup, Mindlife-Ultramind
- **telecommunications:** Vyyo, MRV-Jolt, One Path Networks - Foxcom
- **VLSI CPU design:** DSPG, Fourfold
- **factory automation:** DEC & Iscar, Newport-Optimet, Orisol
- **3D printing**: Cubital
- **radar:** IAI-Elta & DSI
- **automated testing:** Elbit-Elop, Cisco-NDS, Vyyo, DSPG
- **sysadmin:** Cubital, John Bryce, Mitre Corp

I have worked for a number of organizations, large and small, and helped them realize improvements in their product performance, often putting them in the front rank in their field. I have acquired expert knowledge in a number of fields, often liaising with noted experts, and have been able to quickly apply this knowledge to improve the competitive position of the companies and their products. I have a keen interest in

computer languages and compilers. I have created a number of domain specific languages (DSL) that were instrumental in greatly simplifying seemingly intractable problems.

In order to understand my S/W design principles, see the following appendices (in the full version of my CV):

- Programming Language Preferences and Musings
- Domain Specific Languages
- How To Write Correct, Maintainable, Secure, and Easy-to-Test Code

When hiring a senior engineer for a S/W project that has a duration of more than a few months, the most critical requirements are his level of understanding of core programming and architectural concepts, his creativity and innovation track record, and *not* whether he needs an extra week or two to come up to speed in order to become familiar with the specific programming language and toolchain. For example, note my recent smashing success in the field of bioinformatics, where even though I had absolutely zero background in the fields of bioinformatics, biology, biochemistry, and genetics, still within 3 months (admittedly with the help of an excellent mentor) I invented a solution for the client that saved them from bankruptcy.

2.0 Work Experience

2017-present: Cybersecurity Consultant

@Self-Employed, Jerusalem

Keys: cybersecurity, architect, algorithms, obfuscation, compiler, **C/C++**, javascript, WASM

1. I am developing an obfuscating compiler for C/C++ and for Web Assembly (WASM). Still in stealth mode.
2. I am a security mentor for the Jerusalem Mass Challenge start-up hub.

2011-17: Viaccess-Orca: Security Policy Mngr & Architect: Cybersecurity: OTT Internet Pay TV System

@Viaccess-Orca, Ra'anana - a subsidiary of @Orange France, and @Discretix/Sansa Security, Netanya recently acquired by @ARM: See details.

Keys: cybersecurity, DRM, architect, algorithms, anti-reverse engineering, obfuscation, Hex-Rays, OllyDbg, hacking, LLVM compiler, cryptography, data science, big data, fuzzy logic, **C/C++**, Python, TCL, bash, Java, Android root detection, IOS, Linux, ELF binary editor

The product was an Over-The-Top (OTT) Internet pay TV system. We provided the S/W infrastructure to our customers, the satellite and cable pay TV operators, so they could also provide an OTT service to their subscribers in order that they could try to compete with Netflix. The system was designed for small screen Android and IOS devices, i.e. up to 10 inches. We used DRM to encrypt the content.

I was responsible for security policy and security architecture. I worked closely with the product management and the S/W development team leader in order to specify security requirements along with analyzing their costs and benefits. When I specified the architecture of a feature, typically I wrote a working proof-of-concept on account

of its complexity. And I was responsible for evaluation, purchase, and configuration of third party tools.

At the end of my 6 year tenure there were 40M subscribers, and no security breaches.

1. Our workhorse third party tools were the InterTrust WhiteCryption SCP obfuscating C/C++ compiler and their SKB white box cryptographic library. I worked very closely with the vendor's development team because at the time (2014) of our original purchase their SCP obfuscating compiler was new and still had "teething pains". They were aware that I had built a similar in-house product for NDS, so they were happy to cooperate. Given that it takes 5-10 man-years to create a polished product, and Viaccess-Orca needed the product immediately, and anyway did not have the manpower to build it in-house, therefore the purchase decision was a "no brainer".
2. *In many cases the security features that I designed were very complex because their goal was to generate obfuscated binary code in order to prevent reverse engineering, so first I needed to code a working proof-of-concept before finalizing their specifications*.
3. I designed the anti-reverse engineering and obfuscation programming frameworks and libraries in C/C++. All secure code modules on the device were implemented as native libraries written in C/C++. Typically offline utilites were implemented in Python.
4. One of my more important and challenging hacks was to generate a dynamic share object library (DSO) (i.e. an .so file) that formally exports no symbols - *prima facie* an oxymoron. In fact I used an asynchronous back channel that allowed the DSO to communicate with its caller by using a function declared with the gcc constructor attribute that executes before dlopen() returns.
5. I developed an Android root detection mechanism using fuzzy logic techniques. Once the feature was fine tuned, we had zero false positives, and no known false negatives. Note that the cost of false positives to the operator's customer support mechanism can be exorbitant. The cost of handling a single call can sometimes be higher than the monthly subscription fee. And flooding the operator's customer support with thousands of complaints about false accusations of rooted devices could ruin our relationship with our customer, i.e. the operator. Therefore the algorithm had to err on the side of allowing false negatives, i.e. allowing playback on some rooted devices.
6. Originally the exclusive focus of security was protecting the devices from leaking content and keys, i.e. from being reverse engineered. We relied heavily on the premise that we refused to play on "rooted" Android devices or "jail-broken" IOS devices. But when inexpensive rooted Android devices became ubiquitous in the consumer market, the content providers (i.e. the major studios such as Disney, Sony, Warner Bros., etc.) noted a significant drop in their royalty revenues, so they forced us allow playback on rooted devices too. Therefore additional security had to be implemented on the back-end web servers, e.g. to check whether or not a subscriber downloaded an unusually high number of hours of content, or whether the subscriber had simultaneous downloads from different IP addresses. I designed a secure and efficient data logging system. We logged data in order to better understand how subscribers were using the system, and in order to detect piracy. With tens of millions of subscribers, we collected a huge amount of data. I worked with data scientists to design "big data" collection and analysis techniques. And there was the economic challenge to minimize the communication costs of the data collection program.

2016-16: Part-time: Cybersecurity Consultant: Protection of a Small Business with Extremely High Security Concerns

@Anonymous, Jerusalem: See details.

Keys: cybersecurity, privacy, anonymity, WordPress, static web site, Cloudflare, Windows, Android, Google Docs, Google Drive

2015-15: Part-time: Startup: S/W Architect: Biomed: Gait Monitoring App

@Self-Startup (defunct), Jerusalem: See details.

Keys: biomed, architect, android

2010-11: S/W Architect & Developer: Transportation: Urban Traffic Vehicle Route Guidance Algorithms

@TeleQuest (defunct), Jerusalem: See details.

Keys: urban vehicle route guidance, architect, algorithms, Java, AWS

I designed and implemented algorithms along with a computational infrastructure for urban traffic vehicle route guidance similar to what Waze does today.

2009-09: S/W Architect & Developer: Bioinformatics: Invented Algorithm To Overcome PCR Inhibiition

@Syntezza Molecular Detection (defunct), Jerusalem: See details.

Keys: bioinformatics, PCR, algorithms, architect, mathematical programming, data science, AI, **C**, Python

1. **When I started the project, the client's PCR MRSA detection kit had only a 50% detection rate due to inhibition problems associated with their *preliminary* chemistry that could not separate the DNA from the "noise" resulting from the mucus in the patient nasal samples. This result was grossly unacceptable for any medical test. The investors had lost confidence, and were about to pull out. Within 3 months, in spite of my complete lack of background in bioinformatics and biology and genetics, I improved the test's accuracy to 95%, which was 10% better than their competitors from the pharmaceutical giants. I saved the client from liquidation.**
2. I am in the process of patenting my algorithm.

2004-09: NDS: Cybersecurity Researcher for a CA Satellite Pay TV System

@Cisco-NDS, Jerusalem: See details.

Keys: cybersecurity, business development, DRM, algorithms, cryptography, anti-reverse engineering, obfuscation, hacking, LLVM compiler, VM, QEMU, RPC, automated testing, S/W quality, **C/C++**, TCL, Python, Linux, bash, Win32

I was a researcher who worked on a *potpourri* of fascinating and intellectually challenging projects, primarily related to cybersecurity. For example, I worked on the development of obfuscation techniques and a LLVM C/C++ obfuscating compiler, using virtual machines (VM) to crack cryptographic algorithms and DRMs, and business development research to apply NDS security technology to other industries. My background task was to do C/C++ code security reviews. Typically secure coding is achieved by adhering to best programming practices.

2002-03: S/W Architect & Developer: Accessibility: Invented System to Allow Blind to “See” Sonic Maps

@Virtouch (defunct), Jerusalem: See details.

Keys: accessibility, blind, architect, algorithms, GIS, MapML, HTML, SVG, javascript, XSLT, XML Schema, XSLT, **C**, TCLNDS

1. **I was the inventor and architect of a system that allowed the blind to to “see” geographic maps and digital images that were prepared using industry standard GIS map descriptions such as MapML.**
2. Maps were displayed on a standard HTML browser using HTML, SVG, javascript, XSLT, combined with audio feedback.
3. I decided to use an off-the-shelf graphics tablet and stylus instead of the company’s flagship VTPlayer tactile mouse product because the blind can use a tablet much more effectively to navigate the screen compared to a mouse. (Today, 2017, a touch tablet device could be used). The blind have their own sense of hand-eye coordination, and intuitively understand the stylus position on the tablet.
4. **I *almost* saved the company from liquidation. My research was awarded a European FP6 grant of $0.5M Euro that required *matching funds*. But the investors refused to put up the matching funds due to the company’s long history of financial failure in the children’s Braille education market using their expensive tactile mouse.**

1999-2002: Vyyo: S/W Mngr & Architect: Network: Embedded & Offline Utilities for a “Wireless” Cable Modem and Router System

@Vyyo (defunct), Jerusalem: See details.

Keys: network, architect, algorithms, SNMP, SNMP-agent, NMS, automated testing, **C**, TCL, embedded

1. I was the architect of the SNMP network management system (NMS), MIB, and embedded SNMP agent.
2. I was the architect of a hybrid IP connection for cable modems where there was no physical cable upstream channel. Instead the upstream channel used a telephone modem (ATA), while the downstream channel used the cable modem. For typical surfing, the effective downstream rate was as fast as a pure cable solution. The company applied for a provisional patent.
3. I greatly improved the efficiency of the laboratory modem speed stress testing by a factor of 10-100 by using a steepest descent search algorithm instead of a binary search algorithm. Reduced testing time per modem from hours to minutes.
4. I designed a *virtual* testing lab with 64K modems and 512K PCs by *multiplexing the physical connections*. The test lab had only 256 *physical* cable modems, and 4 *physical* PCs with 8 network connections each.

2001-01: Part-time: Consultant: Network Management System (NMS) for a Cable Modem & Gateway System

@One Path Networks - Foxcom, Jerusalem

Keys: SNMP, NMS

I performed a one week requirements study in order to select the most appropriate NMS infrastructure. I saved them over $200K compared to their original selection.

1998-99: S/W Architect & Developer: Compiler: GCC Compiler Port for a 128-Core Stack Machine CPU with a FORTH-like Instruction Set

@Fourfold Technologies (defunct), Jerusalem: See details.

Keys: gcc C compiler, architect, algorithms, DSL, **C/C++**, FORTH, LISP, TCL

1997-98: S/W Architect & Developer: Factory Automation: Conoscopic Interferometer Workstation

@Newport-Optimet, Jerusalem: See details.

Keys: measurement workstation, architect, algorithms, DSL, **C**, TCL, OpenGL, Win32, soft real-time

1996-97: Lecturer: Win32 Internals Course

@Mer Group, Jerusalem

Keys: lecturer, Win32, **C**

1995-96: Elop: S/W Architect & Developer: US DOD Mil-Spec Automated Testing: Night Hawk Fire Control System

@Pitkha Outsourcing (defunct), Jerusalem for @Elbit-Elop, Rechovot: See details.

Keys: automated testing, mil-spec, architect, DSL, **C/C++**, lex/yacc BASIC compiler, Win32, soft real-time

1995-95: Michlala College: Lecturer: Introductory University Computer Science Course on Database Theory

@Michlala College Bayit Vegan, Jerusalem

Keys: lecturer, database, SQL

1991-94: DSPG: S/W Architect & Developer: VLSI: Simulator & S/W Toolchain For DSPG PINE CPU

@Pitkha Outsourcing (defunct), Jerusalem for @DSP Group, Givat Shmuel

Keys: VLSI simulator, S/W Development Toochain, architect, algorithms, DSL, **C/C++**, lex/yacc, assembly, Win32

1. I was the S/W architect of a **clock accurate** DSP CPU simulator along with a complete software development toolchain, i.e. a debugger, C compiler, assembler and linker. Note that the system was developed just *before* the GNU Compiler Collection framework had reached maturity, i.e. v2.95.
2. **This system enabled working applications to be developed before the chip became physically available. It reduced application time-to-market by 6-12 months.**
3. The technological breakthrough was my design of a domain specific language (DSL) that described the CPU architecture **including the pipeline**. Implementation was in lex/yacc and C++.

4. The associated DSL compiler automatically generated the source code for the complete toolchain that enabled it be automatically rebuilt within an hour in the face of almost daily changes to the VLSI architecture - especially the pipeline.
5. **And after every change to the architecture, we ran the complete suite of VLSI verification regression tests 100-1000 times faster than the VHDL simulator.**

1989-91: DEC & Iscar: S/W Architect & Developer: Factory Automation: Shop Floor Production Control (SFPC) System: BARI II

@Digital Equipment Corporation (DEC) (defunct), Herzliya for @Iscar, Tefen: See details.

Keys: factory automation SFPC, architect, algorithms, DSL, Pascal, SQL, VAX/VMS

Iscar Matkash in Tefen IL is a fully automated factory that produces thousands of different cutting blades using a sintering process. The raw materials go through many stages of operations. In many cases after undergoing intermediate processing, the partially processed material can still be diverted to multiple final products - similar to stem cells. The factory contains hundreds of automated workstations, stands, stacks, guided vehicles, and conveyor belts. The product or intermediate product is placed on pallets. The pallets are moved from one stand on a workstation to a stand on another workstation, or temporarily to a storage stand or stack.

My task was to create a computer program that automatically operated/orchestrated the factory. After 18 calendar months, and 6 man-years later, the factory ran perfectly!

1. When my co-architect and I started this project, we had zero background in industrial engineering. We were supplied with a mentor who brought us up to speed.
2. Eventually after months of discussions we created an architecture that was a textbook object oriented taxonomy - a “factory object kingdom”. The top level object was a “production instruction”.
3. We created a descriptive, i.e. *non-procedural*, domain specific language (DSL) that was designed to be user-friendly for the factory engineer.
4. I wrote the language manual.
5. We used the language to configure the factory. We created a GUI which emitted CLI script. But major updates to the database were implemented via very large CLI scripts of tens of thousands of lines.
6. We mapped the language to a relational database.

1988-88: Part-time: S/W Architect & Developer: Accessibility: Quadriplegic PC Accessibility

@Cubital (defunct), Herzliya - a charity project funded by the company and their CEO Itzhak Pomerantz in cooperation with the Lowenstein Rehabilitation Hospital, and the IDF Rehabilitation Unit: See details.

Keys: accessibility, Prolog, PC-DOS

1. First of all, it important to note that this project was done in 1988 when speech recognition technology was still in its infancy, and exorbitantly expensive.
2. **This system was used to enable Shulamit Gabbai, a former school teacher who became quadriplegic by contacting polio (due to a terrible malfunction in the *Or Akiva* drinking water supply which became mixed**

with sewage), to become a book editor for *Maariv*. She was able to type 30 characters per minute on her PC by using a sip-and-puff switch along with telescopically enhanced light pen attached to her head with a woman's plastic hair head band.

1987-88: Cubital: S/W Developer & VAX/VMS Sysadmin: 3D Printer: Solider

@Cubital (defunct), Herzliya: See details.

Keys: 3D printing, **C**, sysadmin, VAX/VMS

1986-87: S/W Developer: Soft Real-Time RS232 Z80 Communication Driver: Data Collection & Access Control Terminal

@Elde (defunct), Jerusalem

Keys: data collection terminal, **C**, RS232, Z80, embedded, real-time

1985-86: Orisol: S/W Developer: Factory Automation: Leather Sewing Workstation

@Orisol, Lod: See details.

Keys: sewing workstation, DSL, algorithms, AutoCad, **C**, awk, PC-DOS

1983-84: IAI-Elta: S/W Developer: Real-Time: Data Collection Terminal & Radar for Lavi Fighter Plane

@DSI (defunct), Givatayim for @IAI-Elta, Ashdod: See details.

Keys: data collection terminal, PL/M, 8080, RTOS, fighter plane radar, Jovial, embedded, real-time

1981-83: Mitre Corp: S/W Developer & IBM CP/CMS Assistant Sysadmin

@Mitre Corp, McLean VA: See details.

Keys: APL, PL/1, sysadmin, IBM CP/CMS

1979-80: Programmer & Economist

@JWWA.com, a regulatory economics consulting firm in the Washington DC area: See details.

Keys: electric utility economics, Fortran, IBM MVS

1977-78: Intervenor/Economist

@Ontario Energy Board (OEB), Toronto: See details.

Keys: electric utility economics

1. I was an intervenor at the ECAP'77 costing and pricing hearings on the subject of introducing electric utility tariffs based upon marginal cost pricing (MCP) (= peak load or time-of-day pricing).
2. **I argued my position very well. At 22 years old, I was the first public interest intervenor in the history of the OEB to be awarded costs.**
3. **I published an op-ed in The Globe and Mail, i.e. at the time Canada's newspaper of record, explaining the economic and political issues surrounding the case.**

3.0 Education

3.1 Formal Education

1979: York University, Canada: MA Economics & Applied Mathematics

See details.

1977: University of Toronto - Rotman School of Management (MBA Program): No Degree

See details.

1976: University of Toronto: BA Economics & Applied Mathematics

The most memorable and still useful courses I took were in statistics, experimental design, game theory, advanced calculus, and microeconomics.

In 1971 at the age of 15, for a high school computer science course, I wrote a computer program to play a perfect game of 3D 4x4x4 tic-tac-toe in Fortran on an IBM 1130. The computer had 16 KB RAM, and was the size of a refrigerator. It was arguably my most formative learning experience from which I received the computer programming "bug" which I carry with me to the present day.

3.2 Continuing Education

Today the field of computer science is changing so rapidly that one's formal education has a half-life of less than 5 years. Therefore in order to maintain my state-of-the-art professional edge, I am involved in an intensive effort of continuing education. See details.

4.0 Spoken Languages

1. English (5/5)
2. Hebrew (4/5)
3. French (2/5)

5.0 Computer Languages, SDKs, and Operating Systems

Language knowledge in order of expertise, *based upon my current frequency of usage*:

1. C, TCL, bash + posix text utilities, e.g. awk, sed, etc.
2. C++, python, make, html5, css, markdown, pandoc, jinja2
3. java, ant, cmake, javascript, yaml, json, flex, bison, llvm
4. go, forth, lisp, prolog, apl, fortran, opengl, svg, xml schema, relax ng, xslt, perl, C#

Note that I write compilers and Domain Specific Languages (DSL), so learning a new language takes me only a few days.

O/S knowledge in order of expertise, based upon my current frequency of usage:

1. Linux
2. Android
3. Win32

4. IOS

6.0 Patents Under Development

- Bioinformatics: (a) An extremely accurate and simple noise reduction and normalization algorithm to improve the accuracy of the standard PCR Ct calculation, and (b) an Artificial Intelligence (AI) methodology for measuring the quantity of DNA in a bioassay where inhibition makes it impossible to estimate the Ct because no underlying logistic function (= a flat “S” shaped curve) exists.

- Cryptography: A set of non-linear cryptographic primitives using Hamming weight-like data dependent permutations which overcomes the well known limitation of using Hamming weights because they have a binomial distribution.

7.0 Personal

I was born in Canada in 1956. I have lived in Jerusalem Israel since 1983. I am married with 4 children, 2B + 2G, plus many grandchildren. I take physical fitness seriously. Once upon a time I was a judoka, and a classical guitarist. I was an IDF reserve soldier for 15 years, where I served as a combat soldier in the infantry in the Jordan Valley. In spite of the fact that I joined the army when I was 32 years old (Hebrew: *Shlav Betnik*), functionally, but unofficially, I served in the capacity of deputy company commander (Hebrew: *Samech Mem Pe*) which provided me with the opportunity to achieve rich personal growth, and enabled me to learn important managerial and leadership skills.