Ivan Gaydardzhiev

github.com/gaidardzhiev www.linkedin.com/in/ivan-gaydardzhiev-3282531ba
1v4n@protonmail.com +30 699 889 4083

Education

Harvard CS50: Introduction to Computer Science *July 2025*
Harvard University in Cambridge, Massachusetts, United States
Electronic Certificate: `https://cs50.harvard.edu/certificates/0a4e16b5-3ca8-4404-a34f-c81a94b3779d`

Elastic Engineer *July 2025*
Elasticsearch Training Program
Certificate ID: `C131225`

LFS101: Introduction to Linux *July 2025*
The Linux Foundation
Certificate ID: `LF-sd75vzpzz9` Credential ID: `e894cac9-abdd-47fd-a963-22f5d78f7361`
Electronic Badge: `https://www.credly.com/badges/e894cac9-abdd-47fd-a963-22f5d78f7361`

LFC108: Cybersecurity Essentials *July 2025*
The Linux Foundation
Certificate ID: `LF-vuxlu4yniw` Credential ID: `528a600a-7816-4a29-a22a-167b17df4f66`
Electronic Badge: `https://www.credly.com/badges/528a600a-7816-4a29-a22a-167b17df4f66`

LFD103: Linux Kernel Development *July 2025*
The Linux Foundation
Certificate ID: `LF-nailp6wsqp` Credential ID: `b0192726-86e1-4c7a-b577-5da5641a26a0`
Electronic Badge: `https://www.credly.com/badges/b0192726-86e1-4c7a-b577-5da5641a26a0`

LFD110: RISC-V *August 2025*
The Linux Foundation
Certificate ID: `LF-6p3ks6n6ug` Credential ID: `e27b5dd6-f218-4285-906b-82f5bc9ea3a7`
Electronic Badge: `https://www.credly.com/badges/e27b5dd6-f218-4285-906b-82f5bc9ea3a7`

LFS169: GitOps *August 2025*
The Linux Foundation
Certificate ID: `LF-13j7k19kc3` Credential ID: `76633aef-434b-49c4-a35a-35830e3c1261`
Electronic Badge: `https://www.credly.com/badges/76633aef-434b-49c4-a35a-35830e3c1261`

LFS158: Kubernetes *August 2025*
The Linux Foundation
Certificate ID: `LF-wf25gd6qf3` Credential ID: `f1d504d1-b4cb-47ad-b608-112567e868b7`
Electronic Badge: `https://www.credly.com/badges/f1d504d1-b4cb-47ad-b608-112567e868b7`

Programming Languages & Technologies

Languages:	C, POSIX Shell, Makefile, ARMv8L Assembly, x86 Assembly, X86-64 Assembly, LaTeX, Groff
Operating Systems:	Linux, FreeBSD, OpenBSD, Plan9
Compilers and Toolchains:	GCC, TCC, Clang, musl-gcc, glibc, uClibc, dietlibc, avr-libc, GDB, GNU Make
Tools:	git, zsh, bash, nmap, ncat, ssh, awk, grep, sed, curl, wget, dig, host, nslookup, fsck, gpg, grub, gzip, masscan, openvpn, openssl, qemu, rsync, systemd, runit, tar, diff, pacman, nc, aircrack-ng, wireshark, bettercap, metasploit, kismet, hping, hashcat, tcpdump, responder, dsniff, ettercap, john, airgeddon

Projects

Toolbox (C, Makefile) *Git: github.com/gaidardzhiev/toolbox*
Developed a compact, statically linked binary in C that consolidates numerous common *nix command line utilities into a single executable. This minimalist toolkit provides essential *nix commands through one unified binary, significantly simplifying deployment and usage. The static linking approach ensures all dependencies are embedded within the executable, eliminating external library requirements and enhancing portability across compatible systems.

Syscall (C, ARMv8L 32bit assembly) *Git: github.com/gaidardzhiev/syscall*
Developed a suite of low level Linux command line utilities and a minimalist shell using only direct system calls invoked via inline assembly, completely bypassing the standard C library. This project involved writing ARMv8L 32bit assembly code, manually handling process initialization, and implementing the command line tools by interfacing directly with the Linux kernel through raw syscalls, demonstrating deep expertise in low level systems programming, ARM calling conventions, syscall ABI, assembly level process control and OS internals.

Crt0trust (ARMv8L 32bit assembly, C) *Git: github.com/gaidardzhiev/crt0trust*
This project presents a proof of concept implementation of the "Trusting Trust" attack, originally described by Ken Thompson. It involves embedding a backdoor in the low level C runtime startup assembly code (crt0.s), which silently launches a reverse shell as soon as a program is executed. The demonstration shows how a compromised toolchain can insert malicious code that remains undetectable in the source code and replicates itself during compilation. It underscores the serious security threat of relying on compiled binaries and compilers without thoroughly verifying the integrity of the entire build process, illustrating the stealthy and self propagating nature of the attack first introduced in Thompson's influential 1984 Turing Award lecture.

SBPM (POSIX Shell) *Git: github.com/gaidardzhiev/sbpm*
Developed a lightweight, POSIX compliant shell script package manager that handles source and binary packages with minimal dependencies (only POSIX shell and wget). It supports bootstrapping native compilers and cross compilers, automates Linux kernel builds, and enables reproducible, portable software builds across multiple architectures, while also managing a wide range of packages including libraries, shells, networking tools, system utilities, window managers, bootloaders, emulators, and telecommunications software making it a versatile solution for diverse development needs.

Scripts (POSIX Shell) *Git: github.com/gaidardzhiev/scripts*
Developed and maintained a comprehensive suite of over 50 custom POSIX compliant shell scripts designed to automate and enhance various system administration, development, and security tasks on *nix operating systems. This extensive script collection has significantly reduced manual effort, improved system reliability, and accelerated development cycles. The modular design allows easy adaptation and reuse across projects, demonstrating strong problem solving and automation skills.

Oldbox (C, Makefile) *Git: github.com/gaidardzhiev/oldbox*
Developed a robust collection of *nix command line tools focused on file manipulation, text processing, system monitoring, process control and logic. The project involved implementing core functionalities such as directory traversal, file reading and writing, string transformation, and inter process communication. Emphasis was placed on creating efficient, reusable code with automated build processes to ensure reliability and maintainability. This work deepened expertise in low level system programming, shell integration, and the design of modular utilities that interact seamlessly within a *nix environment.

Getprand (ARMv7 Assembly, Makefile) *Git: github.com/gaidardzhiev/getprand*
Developed an ARMv7 32bit pseudo random string generator entirely in assembly. It directly invokes Linux kernel syscalls without any libc or runtime, demonstrating deep understanding of low level ARM syscall conventions, memory management, and minimalistic assembly coding.