



AUGUST 7-8, 2024
MANDALAY BAY/LAS VEGAS

Nimplant

A light-weight first-stage implant and C2
written in Python, Nim, **and Rust**

Cas van Cooten

/usr/bin/id

- Offensive Security enthusiast, Red Team operator & self-proclaimed “Malware Linguist”
- Likes building offensive tooling in modern languages (💖 Rust, Nim, Go & Python)
- Publishes OST and various offsec-related repositories on Github
- Semi-pro shitposter on Twitter



Cas van Cooten

 casvancooten.com

 @chvancooten

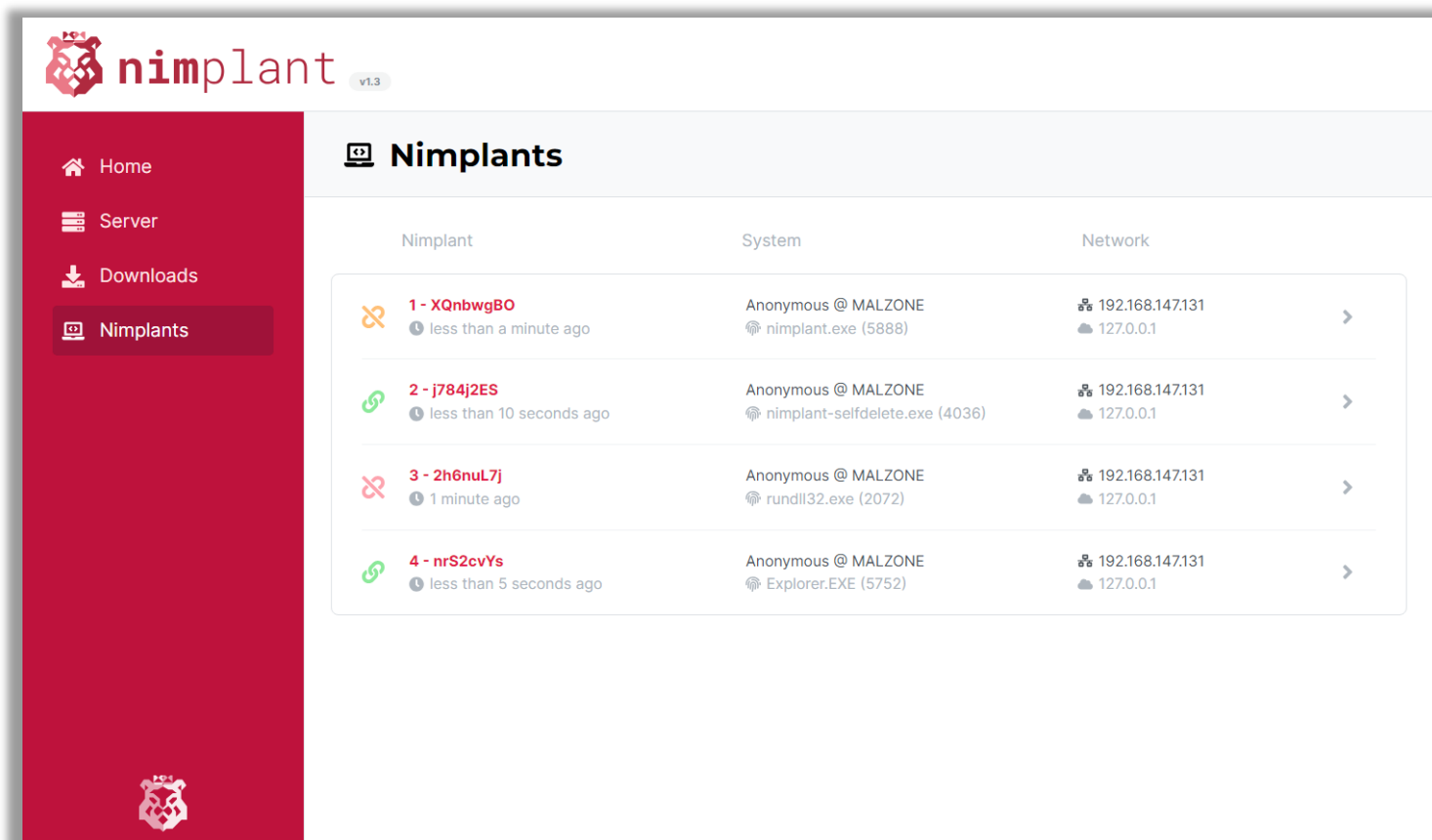
 chvancooten

 /in/chvancooten

Nimplant

A light-weight, first-stage C2 implant

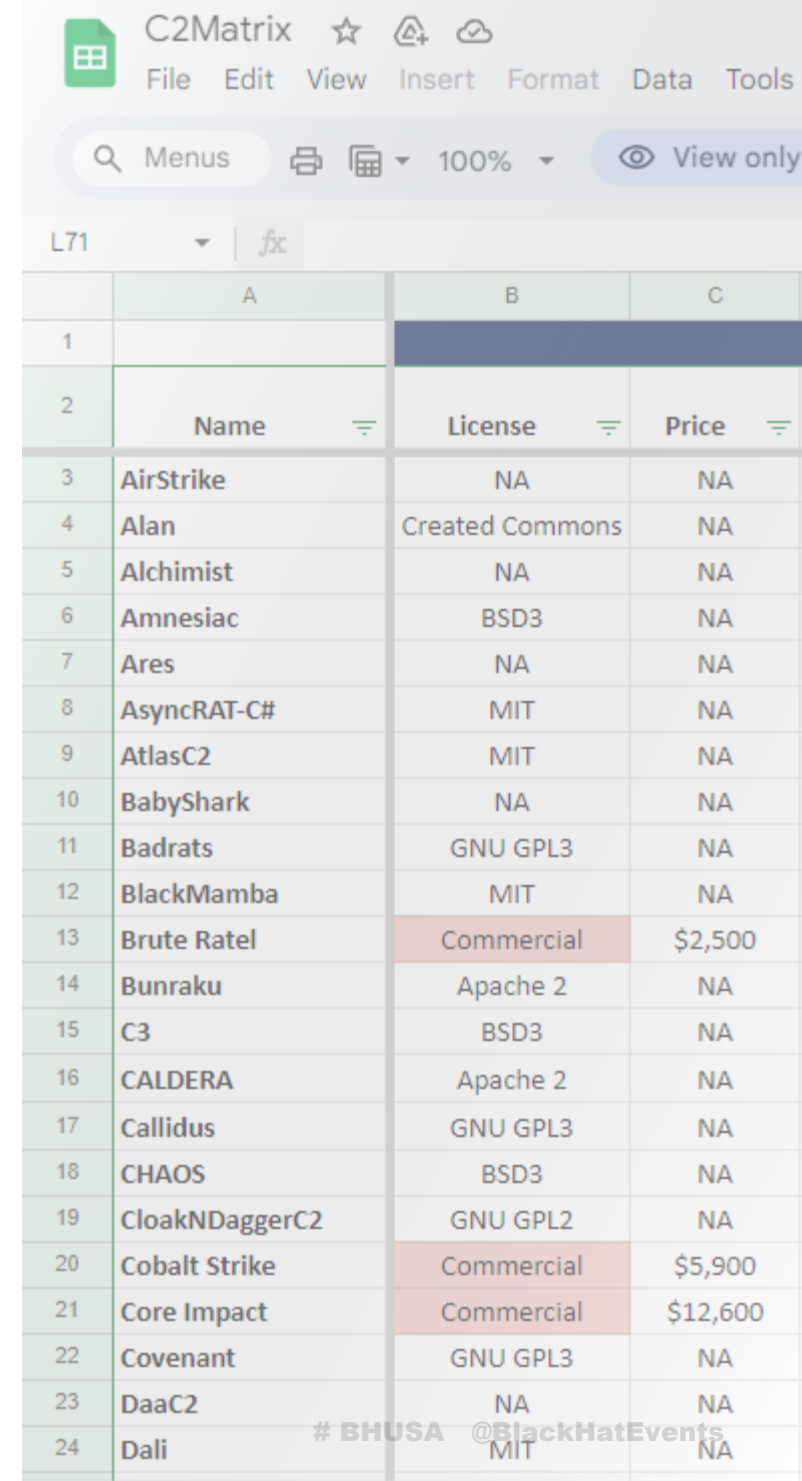
- Started as a side project
- Focus on lightweight footprint and functionality for early operations
- Easy to use and configure, evasive by default (if configured right)
- Fully open-source (MIT):
<https://github.com/chvancooten/nimplant>



Nimplant

But why?

- Started as a side project to learn about development in various languages
- Open-sourced to educate and inform
- **Responsible open-sourcing:**
Public repository is 'defanged', defenders get early access, and detection guidance is provided

A screenshot of a Google Sheet titled "C2Matrix". The sheet has a menu bar with "File", "Edit", "View", "Insert", "Format", "Data", and "Tools". Below the menu bar is a search bar with the text "Menus" and a magnifying glass icon. To the right of the search bar are icons for printing, a grid, and a percentage set to "100%". Further right is a "View only" button with an eye icon. The sheet itself has a column header row with "A", "B", and "C". The rows are numbered 1 through 24. The data in the sheet is as follows:

	A	B	C
1			
2	Name	License	Price
3	AirStrike	NA	NA
4	Alan	Created Commons	NA
5	Alchemist	NA	NA
6	Amnesiac	BSD3	NA
7	Ares	NA	NA
8	AsyncRAT-C#	MIT	NA
9	AtlasC2	MIT	NA
10	BabyShark	NA	NA
11	Badrats	GNU GPL3	NA
12	BlackMamba	MIT	NA
13	Brute Ratel	Commercial	\$2,500
14	Bunraku	Apache 2	NA
15	C3	BSD3	NA
16	CALDERA	Apache 2	NA
17	Callidus	GNU GPL3	NA
18	CHAOS	BSD3	NA
19	CloakNDaggerC2	GNU GPL2	NA
20	Cobalt Strike	Commercial	\$5,900
21	Core Impact	Commercial	\$12,600
22	Covenant	GNU GPL3	NA
23	DaaC2	NA	NA
24	Dali	MIT	NA

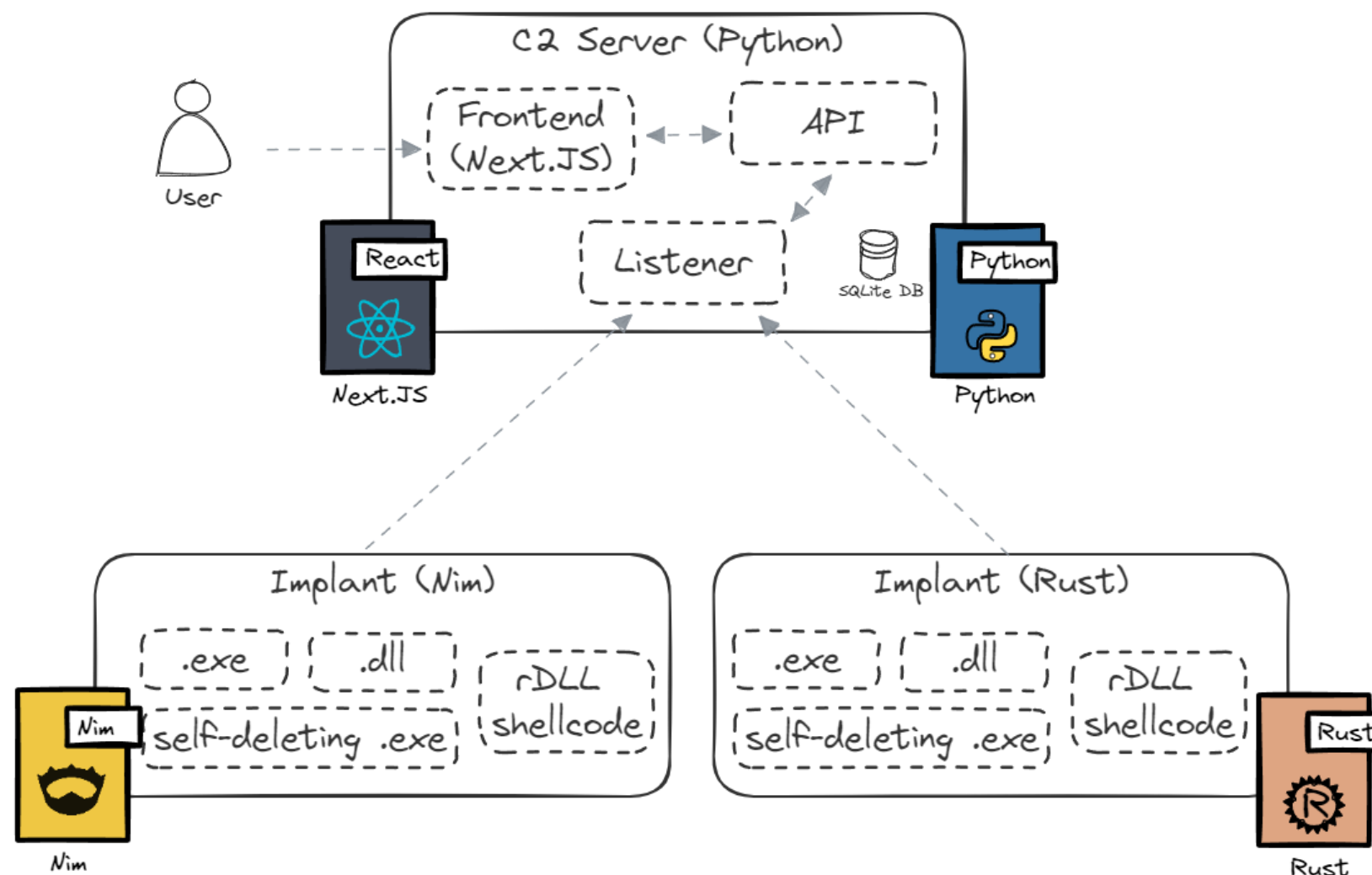
Nimplant

How does it work?

- Simple architecture, consisting of frontend, server, and implant(s)
- Various compilation modes

New for Black Hat 2024!

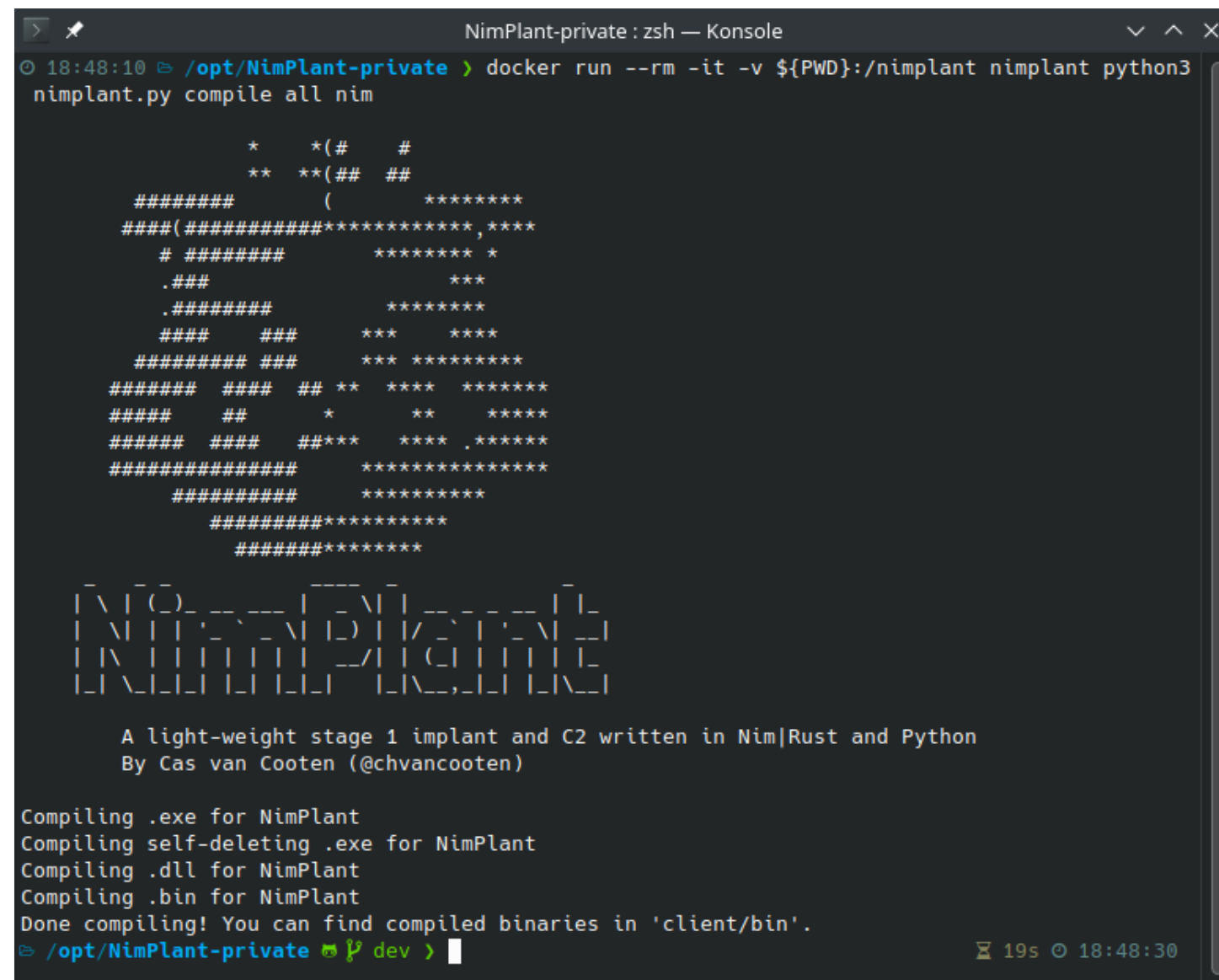
- Rust implant 🦀
- Dockerized setup 🐳



Nimplant

Getting started

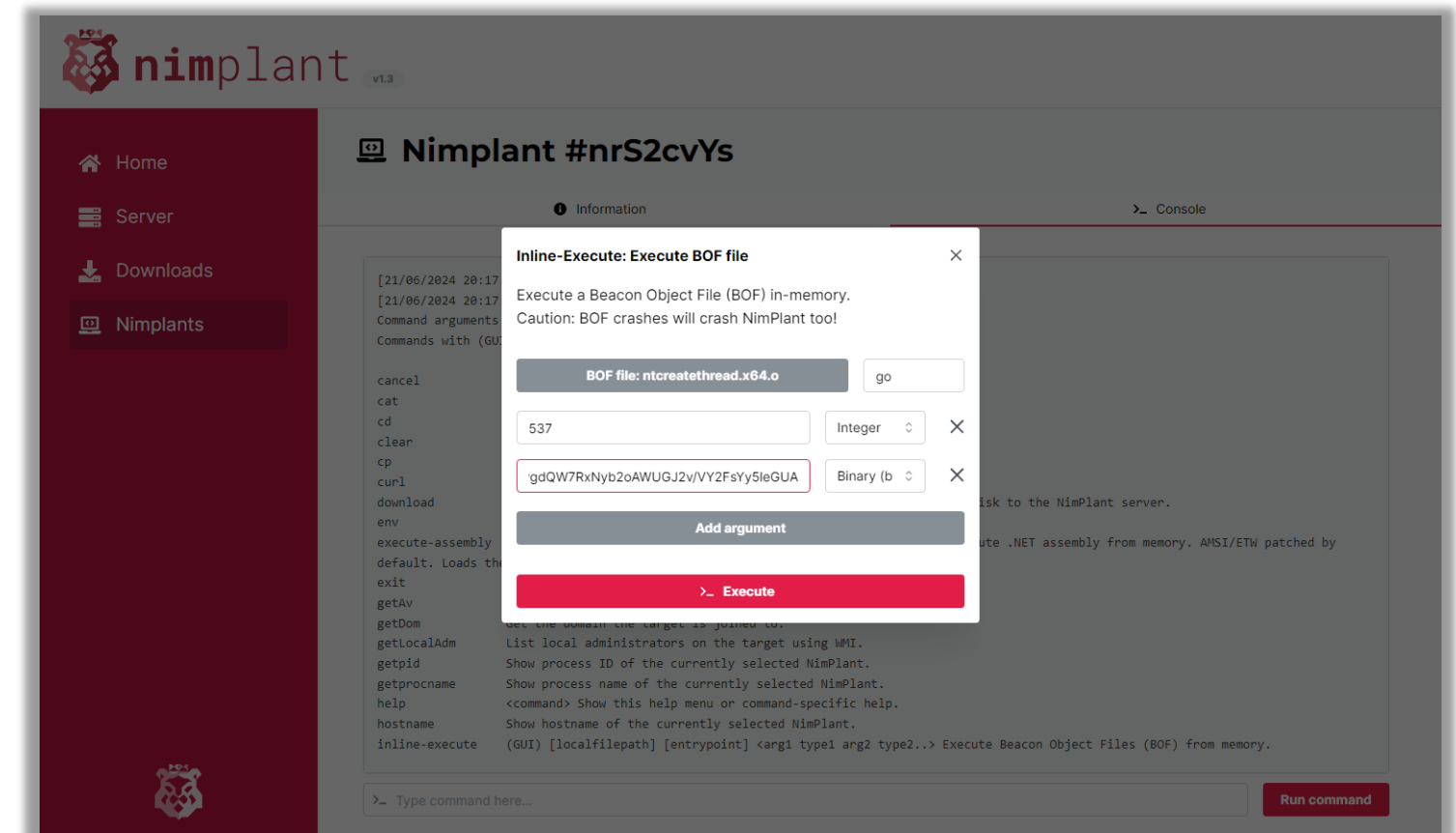
- Configure server and implant via `config.toml`
- Compile desired artefacts locally or via Docker
- Run server locally or via Docker
- Good to go!

A screenshot of a terminal window titled "NimPlant-private : zsh — Konsole". The terminal shows the command `docker run --rm -it -v ${PWD}:/nimplant nimplant python3 nimplant.py compile all nim` being executed. The output displays a large ASCII art logo for Nimplant, followed by a description: "A light-weight stage 1 implant and C2 written in Nim|Rust and Python By Cas van Cooten (@chvancooten)". Below this, it lists the compilation steps: "Compiling .exe for NimPlant", "Compiling self-deleting .exe for NimPlant", "Compiling .dll for NimPlant", and "Compiling .bin for NimPlant". It concludes with "Done compiling! You can find compiled binaries in 'client/bin'." The terminal prompt is `/opt/NimPlant-private dev >`. The bottom right corner shows a timer at 19s and the time 18:48:30.

Nimplant

Usage

- List commands with **help**
- Compile in 'risky mode' to use all commands
- Some commands have GUI modal, just type without arguments



Demo

7-8, 2024
BAY/LAS VEGAS