“Writing is the art of cutting words”. I believe this applies well to programming as well. If you write code, remember to cut lines, keep it as simple as possible and avoid redundancy – this should be a continuous goal.

Interview with Felipe Daragon, creator of Syhunt and Huntpad

[Hakin9 Magazine]: Hello Felipe Daragon (Syhunt)! Thank you for agreeing to the interview, we are honored! How have you been doing? Can you tell us something about yourself?

[Felipe]: Many thanks for the invitation! I'm great! It's a bit cold and cloudy right now here. I was coding a SAST* tool last night while listening to music and sometimes pausing to look at some light rain falling outside the office window. For me, this is the best weather to code. It has been a busy couple of weeks, getting deeper into security aspects of web applications that are built using MEAN**. This is the work I've been doing for some time now, as an application security specialist, diving into cutting edge technologies so that I can create and shape the tools that will help protect them against attacks or allow to inspect them like the Sandcat browser and its cousin project Huntpad.

(*) SAST stands for Static Application Security Testing, a testing methodology that analyzes the source code of applications to identify vulnerabilities.

(**) MEAN stands for MongoDB, Express, AngularJS, NodeJS. These are JavaScript based technologies used today to develop complex web applications.

[H9]: Can you tell us more about your project?

[Felipe]: Yup, let me start by properly introducing the Sandcat. Sandcat is a web browser I designed with penetration testing in mind. Because it exposes the details behind websites, users quickly found out that it can also be used for malware hunting, web development and other activities. The browser has been built with Chromium at its core – the same engine that powers the Google Chrome browser, but Sandcat comes with Lua* rather than JavaScript as its extension language, which gives it architectural simplicity and uncanny flexibility. This project recently gave birth to another open source project, called Huntpad, which is a notepad application designed for penetration testing.

(*) Lua which in Portuguese means Moon, is a powerful, fast, lightweight, embeddable scripting language, heavily used in the game industry and security tools such as Wireshark, Snort and nmap.

The language combines simple procedural syntax with powerful data description constructs based on associative arrays and extensible semantics. Being dynamically typed, Lua runs by interpreting bytecode for a register-based virtual machine, and has automatic memory management with incremental garbage collection. Lua is thus ideal for configuration, scripting, and rapid prototyping.

[H9]: Where did the idea of creating your project come from?

[Felipe]: It started with the notion of re-envisioning the user interface of the web application security scanner I developed. I wanted the scanner to be tightly integrated with a custom web browser, making easier not only the execution of automated but manual

penetration testing. The Sandcat browser started then as a parallel project and kept evolving to the point where integrating the application scanner as an extension became possible.

[H9]: What was the most challenging part in creating your project?

[Felipe]: Initially, I was using Trident (IE's engine) and Python as extension language, then Chromium was released in 2008 and a bit later I stumbled upon the Lua language. Integrating and shipping Python as an extension language was not nearly as successful, practical and beneficial as using Lua, which was really a game changer. Developing my own web browser with the features I wanted and needed turned out to be a great way to sharpen my programming skills, master Lua, learn more about the architecture of web browsers and make me comfortable with open source. With so many open source components at its core, making it open source as a whole became a natural step.

[H9]: Did anything surprise you in the process?

[Felipe]: I was amazed with how difficult it was to read cached resources generated and stored by the Chromium library. Cached files don't have their original names, which was something I would expect. Resources were sometimes compressed with zlib, which is something easy to deal with, but I quickly found out that Chromium stores files smaller than 16k in block files which are essentially container files holding many smaller files, and the metadata about the cached files are stored in these container files, all mapped by a binary index file. I had to come up with my own way to read the cache, which worked but before that my mind was blown. LOL

[H9]: Do you think more security professionals should look into learning Lua?

[Felipe]: Definitely, pen-testers want to see tools and functions broken out and able to be used separately, but still be able to work together to complete some task, and this is something that Lua does very well. Lua was built with modularity, extensibility and flexibility in mind, so I think it's an ideal programming language for hackers.

[H9]: What about the feedback from the github community? Does it influence your software?

[Felipe]: I'm always paying attention to feedback from social networks - the ones I like more are discussion threads where users bluntly share their opinions and thoughts like the tool author is not watching. Probably the most funny or valuable comments out there to be read, if you don't get easily offended. LOL

[H9]: Did you ever got feedback that made you think of something you would have never stumbled upon on yourself?

[Felipe]: The kind of feedback I read gave me a confidence that the project was on the right track, which is something very valuable. But when it comes to

innovation, I have been just doing a lot of research and listening to my inner voice and thoughts.

[H9]: Any plans for the future? Are you planning to expand your tool, add new features?

[Felipe]: Today I'm more an improviser than a planner, intuitively choosing what to explore next, but because all my projects are interlinked, sharing the same code base and components, they are constantly evolving together. Major updates to Sandcat and Huntpad are planned for the near future and contributions are always welcome.

[H9]: Do you have any thoughts or experiences you would like to share with our audience? Any good advice?

[Felipe]: When I started coding, over a decade ago, I only cared if the code was going to work. As the time passed, I learned how it was important to keep it clean, organized and componentized. Carlos Drummond de Andrade, a well-known Brazilian poet and writer once said, "Writing is the art of cutting words". I believe this applies well to programming as well. If you write code, remember to cut lines, keep it as simple as possible and avoid redundancy – this should be a continuous goal. Producing rushed code that just works is still a reality for many busy developers out there and the impact often goes beyond stability, code maintenance and readability issues, it becomes harder to secure and manually review the produced code for vulnerabilities and weaknesses.

If you hunt bugs, Huntpad (github.com/felipedaragon/huntpad). and Sandcat (github.com/felipedaragon/sandcat) may help you save your time when performing websec testing. I hope you enjoy the projects!

ABOUT THE CREATOR AND PROJECT

Felipe Daragon

Huntpad: github.com/felipedaragon/huntpad

Sandcat: github.com/felipedaragon/sandcat

Syhunt: http://www.syhunt.com/en/