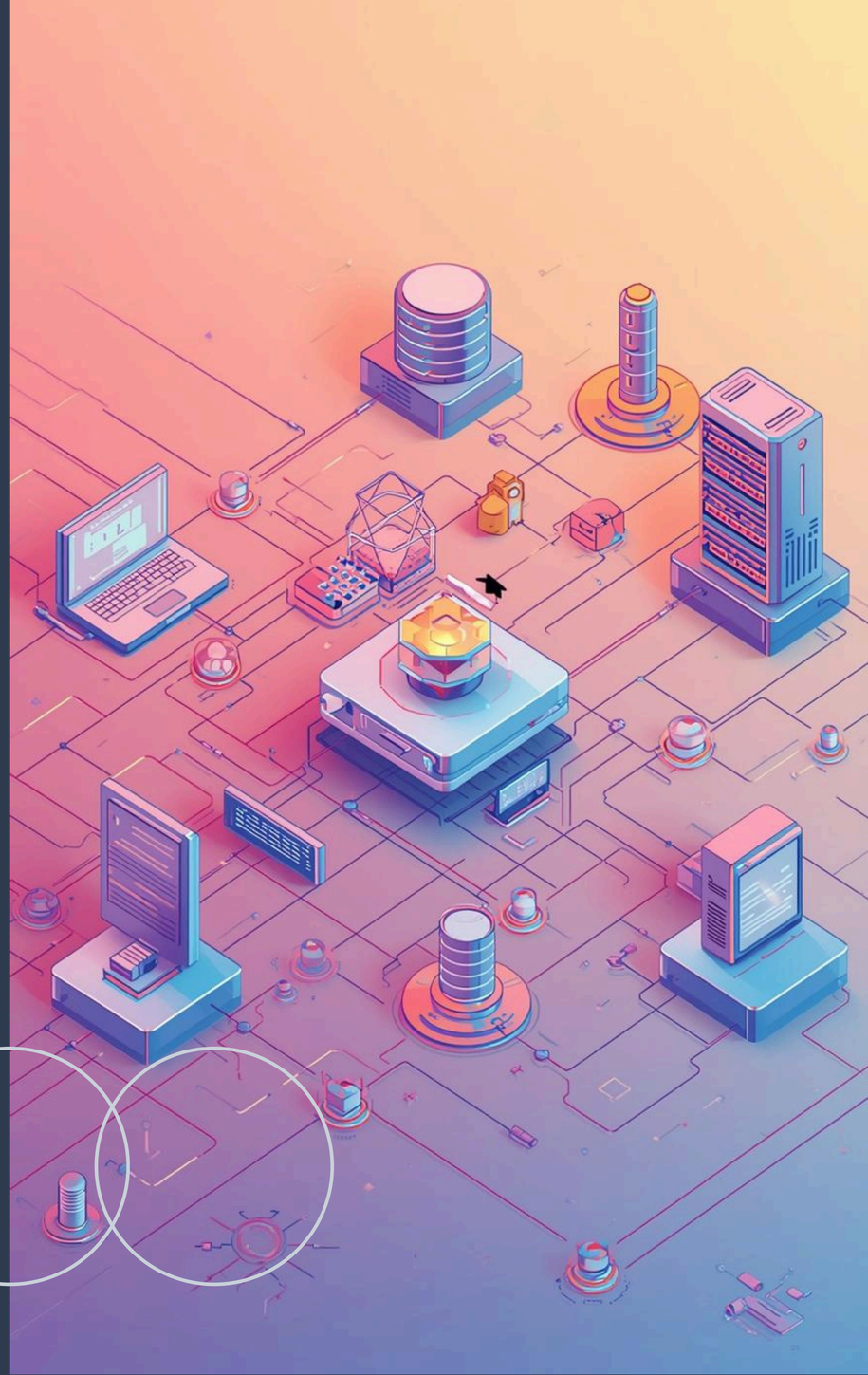
"NEXT GENERATION NETWORKS" COURSEWORK
DISI, UNIVERSITY OF TRENTO

SDN Based HoneyPot



De Toffoli Alessandro

https://github.com/Ale-Deto04/SDN-Based_Honeypot

Introduction

Overview

This project focuses on a **SDN-based security honeypot** architecture implemented on a Kathará-emulated enterprise network.

The scenario models a **company environment** with a legitimate HTTP server, a honeypot, and both trusted and untrusted hosts.

Using SDN capabilities, the controller dynamically **detects** untrusted access attempts and transparently **redirects** malicious traffic to the honeypot, without alerting the attacker.

Introduction

HoneyPot

A honeypot is a **decoy system** designed to attract malicious users by appearing as a legitimate service, with the goal of observing their behavior without exposing real assets.

In this project, the honeypot is implemented as a **dedicated host** that closely replicates the behavior of the legitimate server while providing only fake, non-sensitive information.

This approach allows attackers to interact with a realistic environment, enabling safe monitoring and analysis of malicious activity.

Introduction

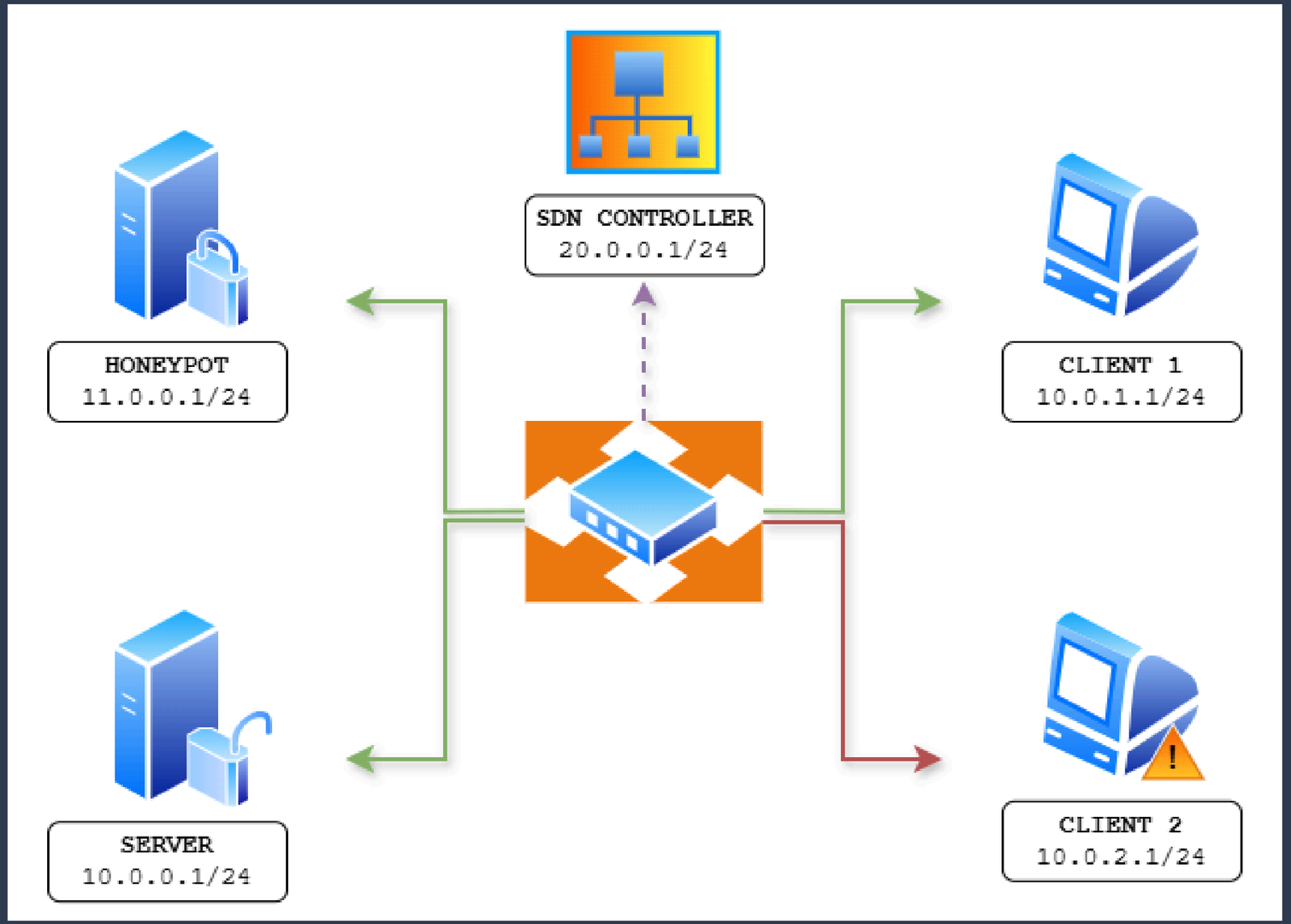
Network

The network is implemented as a **Software-Defined Networking** (SDN) environment composed of multiple subnets connected through an Open vSwitch instance, which acts as the central forwarding device.

An **SDN controller** based on Ryu manages the network behavior, handling routing functionalities, monitoring traffic from suspicious hosts, and performing **Deep Packet Inspection** to analyze application-layer payloads.

Network Topology

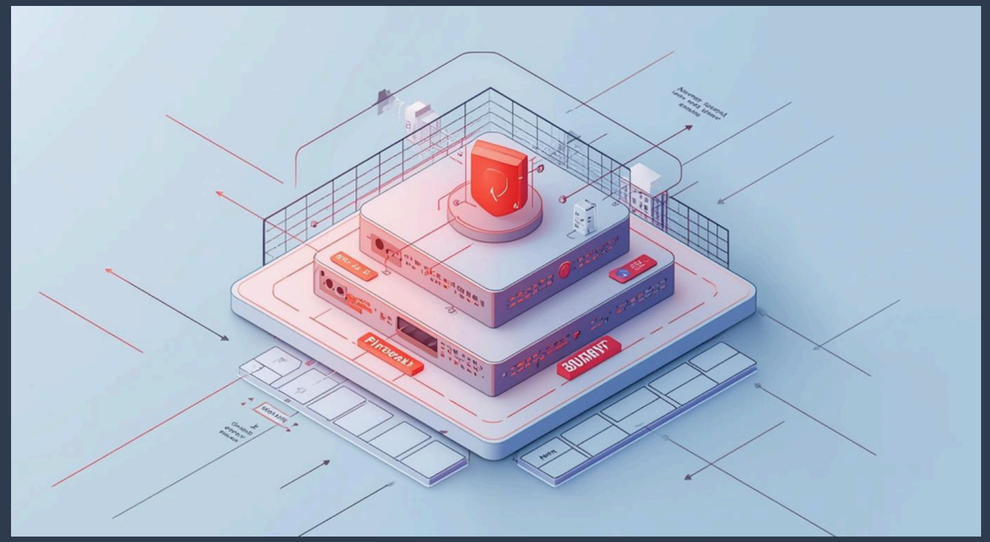
- Four different devices on different subnets
- Open vSwitch linking the collision domains
- SDN Controller administrating the network
- Second client's LAN is untrusted



SDN Controller

Main features

- Routing functionalities
- Deep Packet Inspection on HTTP packets
- Mostly self learning
- Real-time messages and updates to the dashboard GUI



Threat model

Untrusted subnets

Not all company subnets have the same **privileges**.

The server exposes an HTTP service with an **admin-only** area containing private information.

Access to these protected resources is allowed only from the trusted subnet, whereas traffic originating from the untrusted subnet is considered **suspicious** and continuously monitored.

Threat model

Monitoring system

Traffic originating from untrusted networks is continuously monitored by the SDN controller.

For TCP traffic, the controller performs **Deep Packet Inspection** (DPI) on the application-layer payload, searching for specific **patterns** that indicate attempts to access unauthorized or protected resources.

This mechanism allows the system to identify suspicious behaviors at runtime and distinguish legitimate traffic from potential attacks.

Redirection

Installing flow rules

When an intrusion attempt is detected, the SDN controller reacts by **installing a specific flow rule** on the switch.

This rule dynamically **modifies the IP headers** of the packets, redirecting the traffic generated by the suspicious client toward the honeypot.

The redirection is completely transparent to the attacker: from the client's perspective, the communication still appears to be established with the legitimate server, while it is actually handled by the honeypot.

Redirection

Connection loss

During traffic redirection, the legitimate server intentionally simulates a **connection loss**.

This is necessary because the client must establish a new TCP connection with the honeypot, and it is not possible to seamlessly continue the existing TCP session without interruption.

From the client's perspective, the temporary connection failure appears as a **normal network issue**, after which communication is re-established with what is believed to be the original server, but is actually the honeypot.

Redirection

Real-world scenario

In fact, in real-world deployments, honeypots are often not implemented as fully separate servers.

Instead, they are commonly integrated as decoy **portions of the same service** or implemented through **proxy-based architectures** to overcome TCP-level limitations and TLS-related issues.

.

Tools & Technologies

Main technologies involved

Kathara

Kathará is a network emulation framework used to simulate the enterprise network, enabling realistic testing of the SDN-based architecture.

Ryu

Ryu is a Python-based SDN framework that provides libraries for building network controllers. The proposed controller is derived from the simple_switch_13.py.

Flask

Flask is a lightweight Python web framework used to implement the HTTP services in the project, including both the legitimate server and the honeypot.

Demo

GitHub Repository
https://github.com/Ale-Deto04/SDN-Based_Honeypot