

Honeypots in Cybersecurity Strategy From KYGnus

Welcome to our presentation on honeypots, a powerful tool in modern cybersecurity. We'll explore their types, benefits, implementation, and best practices. Join us as we uncover how these deceptive systems can enhance your organization's security posture.





What is a Honeypot?

Decoy system

A honeypot is a controlled, isolated environment designed to attract attackers.

Mimics real assets

It appears vulnerable but is closely monitored to study attack patterns.

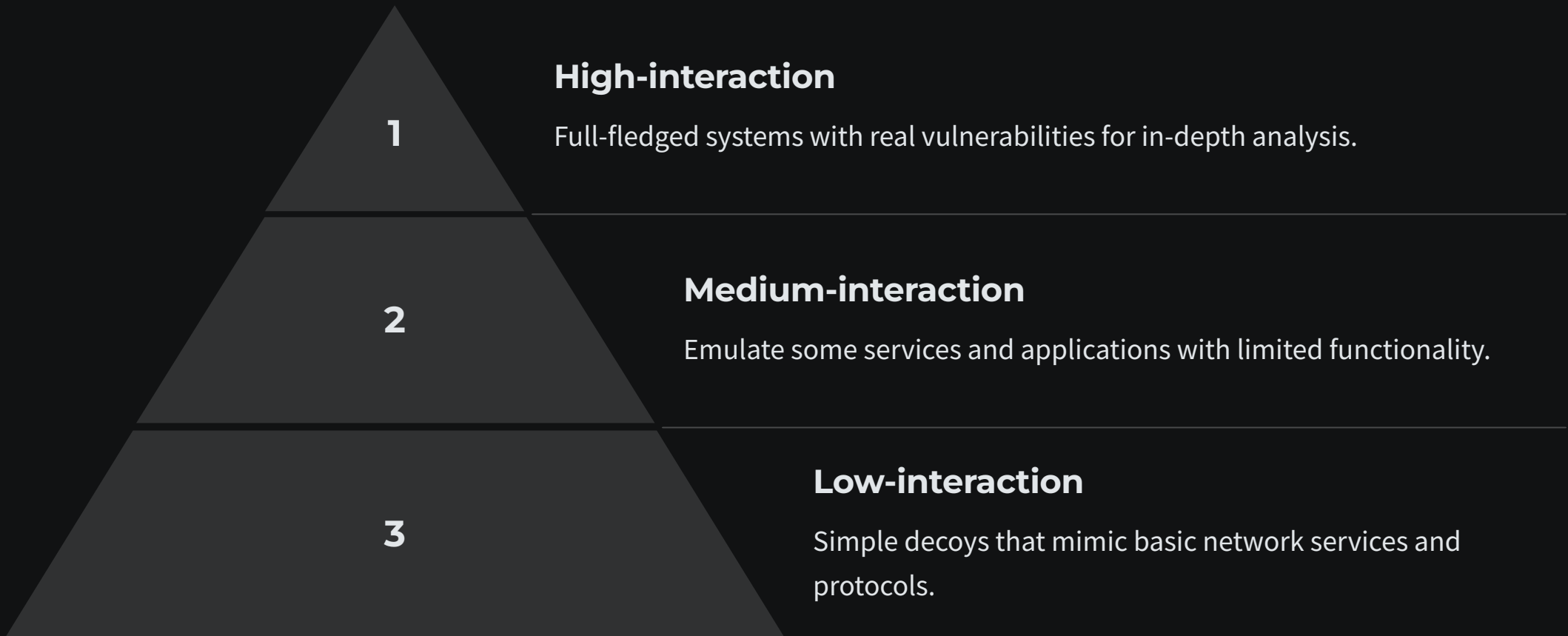
Data collection

Honeypots gather valuable information about attacker tactics and tools.

Early warning

They can serve as an early detection system for network intrusions.

Types of Honeypots



Benefits of Using Honeypots

Enhanced Detection

Honeypots proactively identify novel and unknown attack methods.

Reduced False Positives

Honeypot interactions strongly indicate malicious activity.

Attacker Insight

Gain critical intelligence on attacker tactics and strategies.

Research Opportunities

Honeypots drive the development of cutting-edge security solutions.



Implementing Honeypots

1

Planning

Define objectives and choose the appropriate type of honeypot.

2

Deployment

Set up the honeypot in a controlled, isolated environment.

3

Configuration

Implement monitoring tools and logging mechanisms for data collection.

4

Testing

Verify the honeypot's effectiveness and adjust as necessary.

Integrating Honeypots into Security Operations

1

Placement

Strategically position honeypots within your network architecture.

2

Monitoring

Integrate honeypot alerts with your SIEM system.

3

Response

Develop incident response procedures for honeypot-detected threats.

4

Adaptation

Regularly update honeypots based on new threat intelligence.



Analyzing Honeypot Data

Data Collection

Gather logs, network traffic, and system changes from honeypots.

Pattern Recognition

Identify common attack patterns and techniques used by adversaries.

Threat Intelligence

Use honeypot data to create and update threat intelligence feeds.

Reporting

Generate actionable reports for security teams and management.

Real-World Honeypot Examples

cowrie

Cowrie is a medium to high interaction SSH and Telnet honeypot designed to log brute force attacks and the shell interaction performed by the attacker.

kippo

Kippo is a medium interaction SSH honeypot designed to log brute force attacks and, most importantly, the entire shell interaction performed by the attacker.

ElasticPot

An Elasticsearch Honeypot.

MysqlPot

MySQL honeypot, still very early stage.

pghoney

Low-interaction Postgres Honeypot.

Glastopf

Web Application Honeypot.

Nodepot

NodeJS web application honeypot.

WebTrap

Designed to create deceptive webpages to deceive and redirect attackers away from real websites.

owa-honeypot

A basic flask based Outlook Web Honey pot.

ddospot

NTP, DNS, SSDP, Chargen and generic UDP-based amplification DDoS honeypot.

Best Practices for Effective Honeytrap Deployment

1

Realistic Design

Create convincing decoys that blend with your real systems.

2

Isolation

Maintain strict segmentation between honeypots and production environments.

3

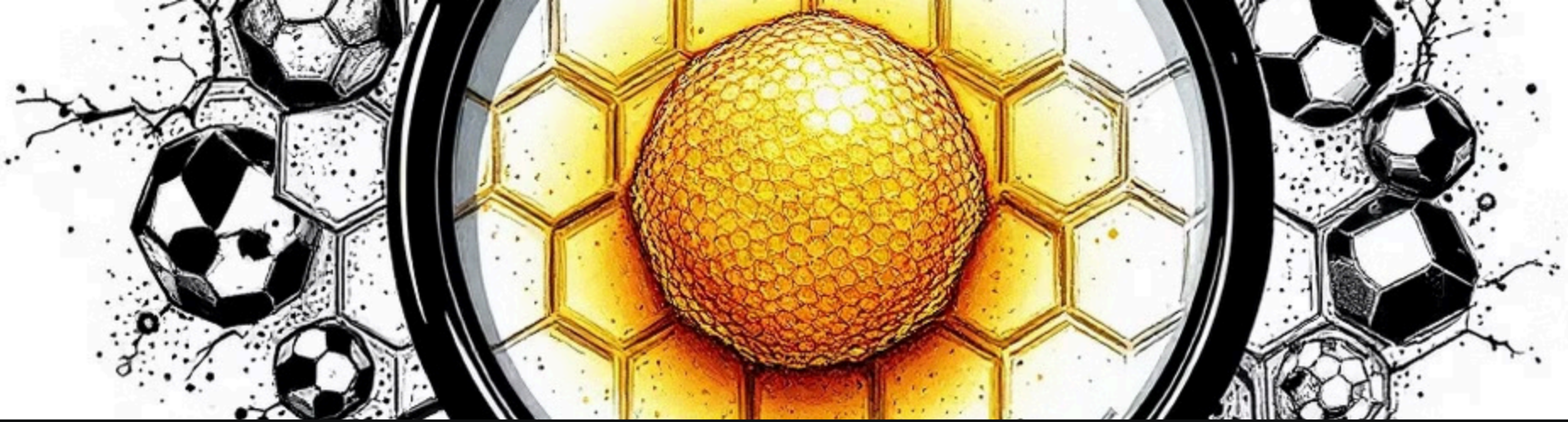
Continuous Monitoring

Implement 24/7 surveillance and automated alert systems.

4

Regular Updates

Keep honeypots current with evolving threats and attack techniques.

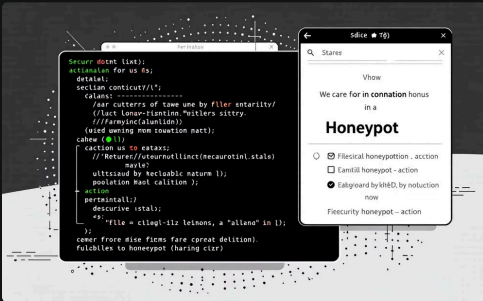


Philosophical review of the honeypot

A philosophical examination of honeypots begins by asking: What is the purpose of installing and running them? Is the goal problem-solving, or simply demonstrating system completeness by deploying one? The goal is to engage the attacker and analyze their behavior. The key question is: does the attacker encounter a familiar system? Is that familiar system, in turn, a honeypot for the attacker? Technically and philosophically, the answer is no.

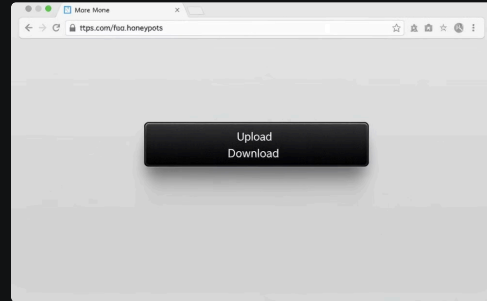
In short, each organization must develop its own honeypot tailored to its specific systems and operational context to maximize its effectiveness against attackers. For enhanced security, Miao utilized diverse honeypot styles—a crucial aspect of maintaining engagement. Own Web Application Honeypot + own Linux Honeypot + Cowrie

Our Honeypot Development



Linux Shell Honeypot

Attempts to solve the flaws of systems such as Cowrie to make analysis more difficult for the attacker.



Web Application Honeypot

Simulates user names and information, allowing download of system information and upload of data for analysis.



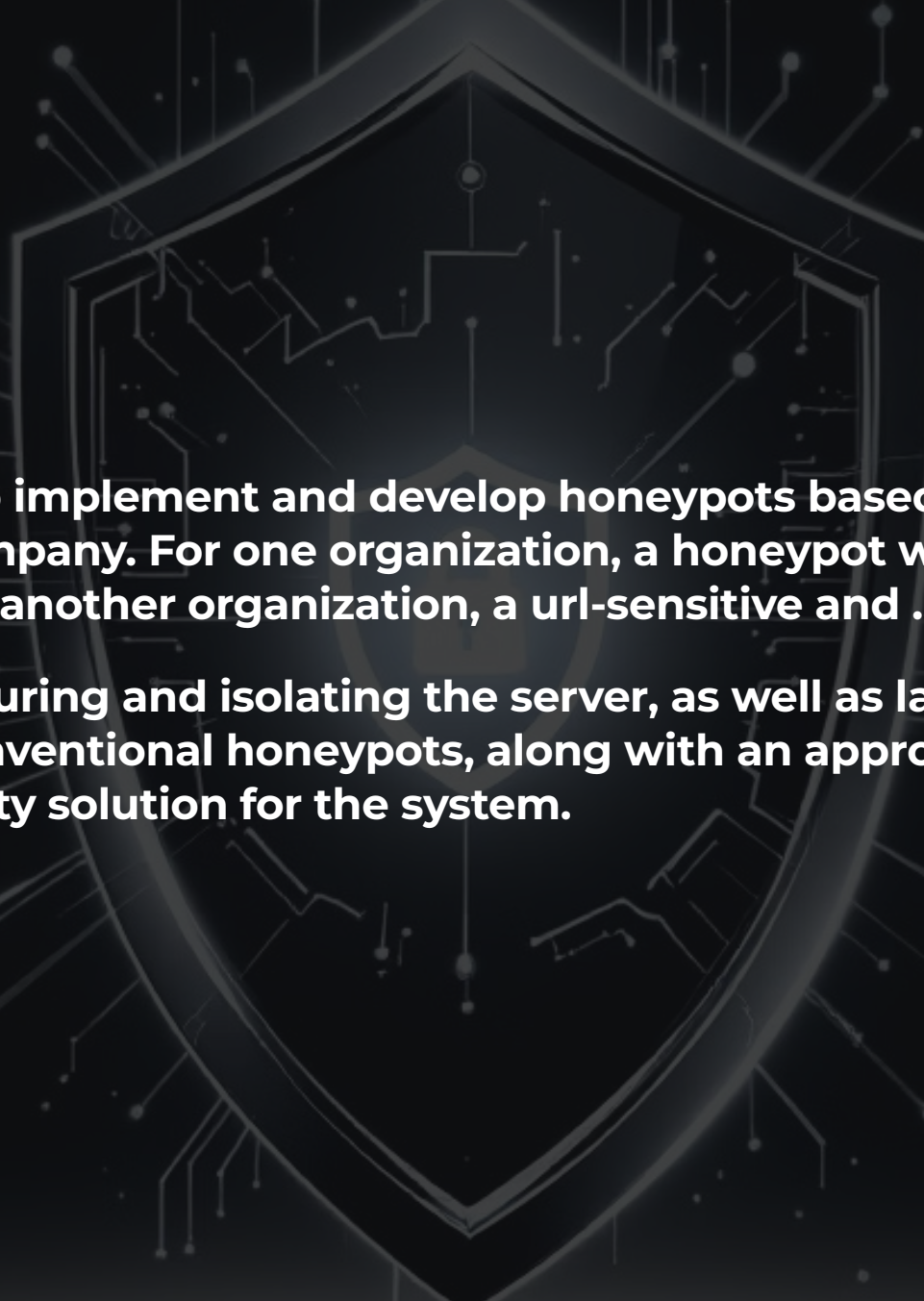
Analyzing Uploaded Files

Examining uploaded files helps determine user purpose and goals.



File Download

file download allows controlled interaction with the attacker.



Our special program is to implement and develop honeypots based on the needs of the organization and the company. For one organization, a honeypot web application with database display and for another organization, a url-sensitive and ...

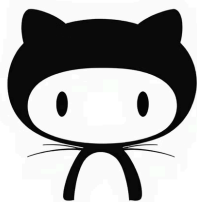
At the same time, configuring and isolating the server, as well as launching the honeypot shell and using other conventional honeypots, along with an appropriate security setup, will provide a new security solution for the system.

About



Website

kooshayeganeh.github.io/



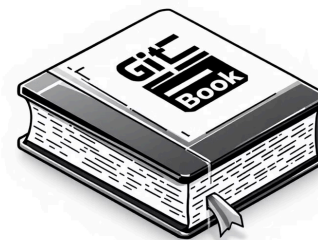
GitHub

github.com/KooshaYeganeh



GitLab

gitlab.com/KooshaYeganeh



GitBook

kooshayeganeh.gitbook.io