Threat intelligence and attack predictions in the ASEAN region
The YAKSHA platform

Alessandro Guarino
Founder & CEO, StAG S.r.l.
YAKSHA Innovation Manager

EEMA Annual Conference 2019 - London
The Speaker

20 Years in Information Security and Data Protection

Speaker / Author

CyCon
International Conference on Cyber Conflict
2013

ISSE
2016 → 2013-2018

XX
Events
2016

IET
The Institution of Engineering and Technology
2017

UNINFO
2011 →

ETSi
CEN/CENELEC/ETSI

enisa
THE EU CYBERSECURITY AGENCY

StAG
INFORMATION GOVERNANCE

NATO
2018

OTAN
2019

12th INTERNATIONAL CONFEREN
CPDP DA

AND

Speaker / Author

CyCon
International Conference on Cyber Conflict
2013

ISSE
2016 → 2013-2018

XX
Events
2016

IET
The Institution of Engineering and Technology
2017

UNINFO
2011 →

ETSi
CEN/CENELEC/ETSI

enisa
THE EU CYBERSECURITY AGENCY

StAG
INFORMATION GOVERNANCE

NATO
2018

OTAN
2019

12th INTERNATIONAL CONFEREN
CPDP DA

AND

Standards and Policy

UNINFO
2011 →

ETSi
CEN/CENELEC/ETSI

enisa
THE EU CYBERSECURITY AGENCY
Rationale

YAKSHA Motivation
Develop and implement a software toolkit to improve Cybersecurity of organisations in the ASEAN region

YAKSHA Results
Enhance cybersecurity readiness levels for its end users, help better prevent cyber-attacks, reduce cyber risks and better govern the whole cybersecurity process.

Process & Strategy
Focus on adapting & integrating other domain technologies into innovative solutions
Overview

YAKSHA is a platform which allows the automated deployment of honeypots, data collection and analysis as well as reporting and information sharing with affiliated YAKSHA installations.

YAKSHA enables organisations, companies and government agencies to deploy custom honeypots meeting their own specifications, monitor attacks in real time and analyse them.
Overview

YAKSHA is a platform which allows the automated deployment of honeypots, data collection and analysis as well as reporting and information sharing with affiliated YAKSHA installations.

YAKSHA enables organisations, companies and government agencies to deploy custom honeypots meeting their own specifications, monitor attacks in real time and analyse them.
Use cases and pilot projects 2019-2020
Features

• **Distributed**: The architecture is inherently distributed. YAKSHA makes possible to deploy easily and cost-effectively hundreds of honeypots through its interconnected nodes. The distributed nature of the YAKSHA system allows also to leverage information and knowledge gathered by nodes outside of one’s organisation, improving its readiness and defensive capabilities.

• **Modular**: It allows both opportunistic and continuous sample collection, as well as selective information sharing with other entities when necessary. Users can upload custom honeypots, monitor attacks in real time and analyse them.

• **Scalable**: It is easy to scale up installations by adding nodes to the network, up to national and international scale.
Features

• **Systems and Tools:** YAKSHA will provide hooks for IoT devices, Android and SCADA systems, as well as regular Windows and Linux. In addition, YAKSHA provides machine learning tools and AI algorithms that can detect malware more accurately, correlate the information with other samples, and extract attack vectors and patterns.

• **Automation:** the platform allows the automated creation of nodes and honeypots deployment, data collection and analysis as well as reporting and information sharing with affiliated YAKSHA installations.

• **Policies:** since honeypots may expose stakeholder’s specific vulnerabilities, each YAKSHA node has the capability of specifying policies for information sharing: the ability to limit the sharing of information outside a single organisation (if the user choses to), as well as anonymization and data protection by default.
YAKSHA is a Honeypot management environment

Honeypot: a machine deliberately designed to provide a wide attack surface and made vulnerable to lure adversaries to attack it.

Strictly monitored in order to collect all interactions made with it, secured.

YAKSHA aims to distil real (threat) intelligence from the raw data.
Two kinds of machines are hosted by the platform. The machines which are used as honeypots are machines which will be used by end-user

1 – VMs to be used as models for custom Honeypots (Windows, Linux, Android, IoT, SCADA)

2 – VMs used for analysis of binaries collected from the honeypots. Not directly accessible to users, but managed by the platform.
Elements

Honeypots Server

Environments: MS Windows, Linux, IoT, SCADA

Conpot for emulating SCADA and IoT

Automated management via custom vagrant scripts developed by YAKSHA.
- start / halt / suspend / delete
- customization of the model VMs to suit the use case

Management of the included the implementation of two distinct web-services, a traditional SOAP-based that uses XML, and another REST-ful API that uses JSON requests and responses. This layer automates the activities relating to single VMs and allows for scalability.
Elements

Infrastructure machines

Constituents of the backend data analysis module: extract information (features), build malware datasets and learn. Cuckoo is one of the elements chosen for this task.

The management/admin system:

User-friendly graphical web interface (Angular) that allows for a variety of workflows focused on creating, configuring, maintaining and destroying a virtual machines. Furthermore, our customized design involves operating and managing computing resources from multiple hosts (including remote or geographically dispersed hosts), while user details and quotas are managed by the YAKSHA admin users with special privileges.
Elements

Data collected

Honeypots VMs are preconfigured to log all traffic, user commands, and filesystem changes in order to collect all the necessary evidence of an attack (forensics preparedness). This information is forwarded to the YAKSHA server and stored in the database.

The data management of YAKSHA is based on two underlying databases, one relational and one NoSQL.

- user related data, policies, reports, VM-related information etc. is stored in a relational database, while
- collected malware data is stored in a NoSQL system (MongoDB)
Architecture overview

YAKSHA Platform
- Analysis
- Reports
- AI, ML

YAKSHA Database

YAKSHA User Login
Registration

CUCKOO Dynamic Analysis

Logs, Network data, commands, Binaries from attackers
Leveraged FOSS Projects

- UBUNTU 16.04 as the OS to host hypervisor provider
- KVM/QEMU as platform hypervisor provider (https://www.linux-kvm.org/)
- CUCKOO as sandbox (https://cuckoosandbox.org/) It was chosen based on its characteristics – it is a sandbox environment which also works on docker. Android sandbox and honeypot environments have been implemented through cuckoo.
  - Cuckoo does not support SCADA and IoT environments hence partners developed the necessary VMs for SCADA / IOT honeypots, collecting data and integration with above installations has taken place through a web service (REST API) supporting Linux, Windows and Android operating systems
- Vagrant (https://www.vagrantup.com/)
- Virtual Box as internal virtualization platform for honey pots VMs (https://www.virtualbox.org)
  - Connecting users to these virtual machines is allowed through SSH (Linux) and WinRM (Windows) and Android so that the end users can remotely install the necessary software that they want and customize their honeypot according to their needs is in progress.
- cloudstack to manage Infrastructure (https://cloudstack.apache.org)
- The YAKSHA database is developed on MongoDB and PostgreSQL is the repository for users data (roles etc) and extracted data from malware + reports generated
- OSSEC (https://www.ossec.net/)
- Fabric (http://www.fabfile.org/)
- Conpot (http://conpot.org/)
- Logstash (https://www.elastic.co/products/logstash)
- Elasticsearch (https://www.elastic.co/products/elasticsearch)
- Kibana (https://www.elastic.co/products/kibana)
Security

Security of YAKSHA honeypots

Not surprisingly, malware developers have started designing malware with the ability to detect whether it is been executed in a VM or sandbox (anti-security and anti-forensics).

**Virtualized systems inevitably leave artifacts in guest OS.** These artifacts include for instance others processes, registry keys and values, loaded and exported DLLs, etc.

In order to harden the virtual machines YAKSHA hides their existence as far as possible by properly configuring the model machines.

Our virtual machines apply state-of-the art methods to prevent fingerprinting and identification from an adversary.
Data Analysis

Automated static analysis

The system identifies the category any file belongs to, and implements the most relevant or appropriate static analysis method.

**Feature extraction**: the system performs this task on the collected malware in order to use them afterwards in the ML/Analysis models.

Features include *static* ones, which may include the existence of specific code snippets, packers, hardcoded strings and URLs and *dynamic* ones, such as URLs, registry changes, filesystem changes etc.

The data collection methodology established a baseline of activities that leads to determining what data **YAKSHA collects** regarding remote interactions and malware analysis, what assumptions, limitations and legal ground are relevant, **what methods and tools to adopt** for data collection, and what reference architecture design is suitable for YAKSHA data collection, management and processing.
Thanks for Your Time
What are your questions?

Contacts:

a.guarino@stagcyber.eu

YAKSHA Project Site
www.yaksha-project.eu

StAG – Information Governance
www.stagcyber.eu

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement Nº 763562.1