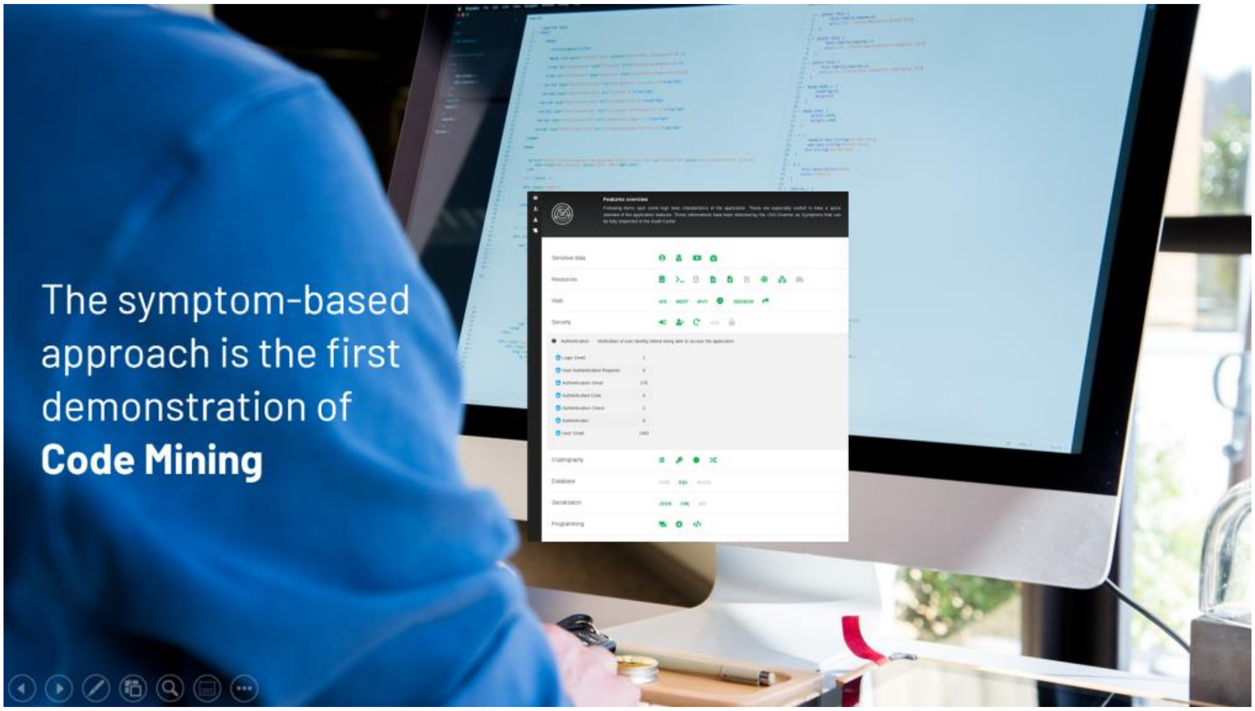


FEBRUARY 2022

The symptom-based approach is the first demonstration of Code Mining

An analogy with the medical approach

We can compare the Code Mining approach with what a doctor does when he receives a patient, correlates symptoms to make a diagnosis and prescribes a medication. Well, Code Mining-enriched analysis follows the same process. It starts by searching and mapping symptoms throughout the source code of the application before making a diagnosis and proposing the accurate remediation to apply.

What is a symptom?

A symptom is not a vulnerability

A symptom is simply a code property. To be precise, it is not a vulnerability but a characteristic of the analyzed application, expressed independently of the programming language.

What is it used for?

Code Mining is used for the mapping of the features of an application

To be more concrete, we are presenting here below an example of what a Code Mining-enriched analysis can be used for:

Example: to facilitate the discovery of an unknown application

As the entire application source code is analyzed, one of the first benefits of symptoms detection is that it enables an exhaustive mapping of all the code's properties of interest, also called "features".

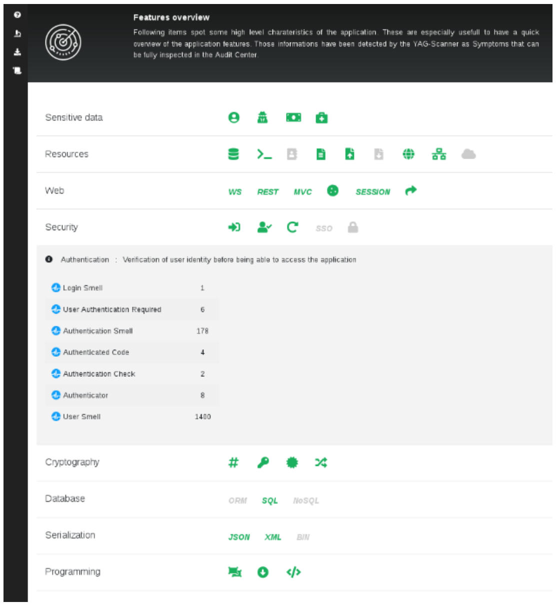
What the user gets is a clear mapping of the features of the application displayed in a structured way.

Symptoms are gathered by category (see screenshot below).

- The first category gathers all the features related to **sensitive data such as personal data handled by the application, secret data, financial data, and health data.**

- The other categories of **interest for security such as access to resources and web services** are also mapped: system calls and network access, web access, etc.

For each category, the complete list of symptoms (in blue) that have been detected is available.



Features overview

Following items spot some high level charateristics of the application. These are especially usefull to have a quick overview of the application features. Those informations have been detected by the YAG-Scanner as Symptoms that can be fully inspected in the Audit Center.

Sensitive data

Resources

Web: WS REST MVC SESSION

Security: SSO

Authentication : Verification of user identity before being able to access the application

Login Smell	1
User Authentication Required	6
Authentication Smell	178
Authenticated Code	4
Authentication Check	2
Authenticator	8
User Smell	1400

Cryptography

Database: ORM SQL NoSQL

Serialization: JSON XML BIN

Programming

As a result, the power of Code Mining enables to uncover symptoms which cannot obviously be detected at first. For example, an "Authentication Smell" type symptom or what experts commonly call a "Code Smell ". (i.e., in this example an area of code that looks like an authentication mechanism).

An automated, exhaustive, and structured code identity form

To summarize, we can say that this complete identity form of the application code has key advantages, such as:

- A very useful tool for the developer who joins a new team

- A helpful mean for the expert or the tierce maintenance applicative specialist (TMA) who starts a new project

It's an automated, exhaustive, and structured source of information **for each AppSec stakeholder** who saves precious time in discovering and taking hands on the application. Ice on the cake: experts are very happy to have statistical elements they can easily integrate into their report whom for?

Do not hesitate to get in touch to get more information
on the Code Mining Code Analysis by Yagaan.

contact@yagaan.com