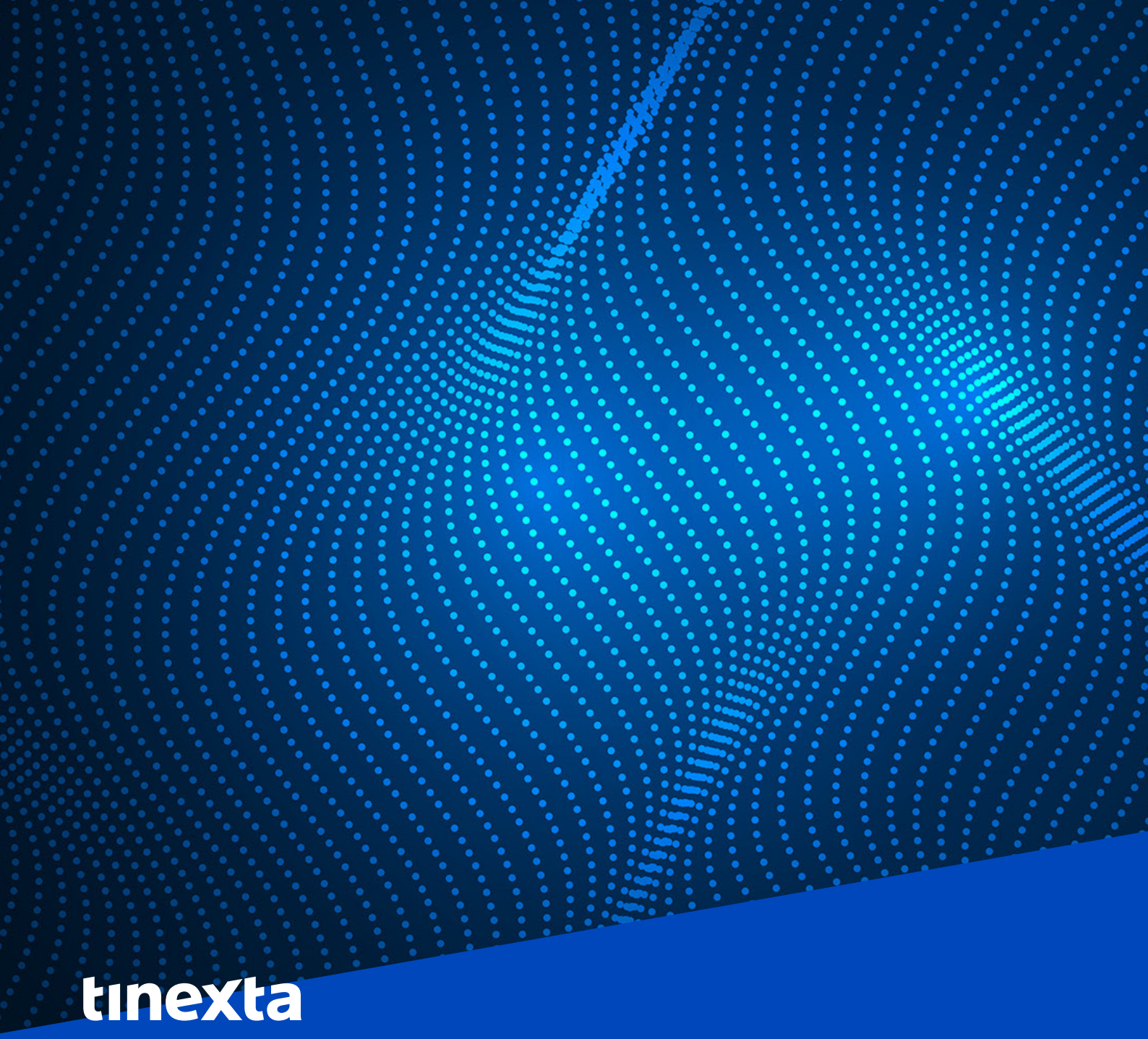


tinexta
defence

Open source release yara-x dotnet

#TinextaDefenceBusiness

Malware Lab

Summary

Our Malware Lab

Tinexta Defence Malware Lab daily performs dissection of malware with the aim of timely understanding the technological evolutions of attacks, consolidating the knowledge of necessary to make more effective and faster the process of incidents responding, contributing to spreading information about emerging threats into the expert's community and among its clients.

Malware Lab analysts are continuously engaged in searching and experimenting new analysis tools, for increasing accuracy and scope of action with regard to the proliferation of new evasion and anti-analysis techniques adopted by malware.

The Malware Lab is also committed to the development of proprietary tools for malware analysis and supporting the management and response of incidents.

Besides malware analysis, Malware Lab ideated and implemented an automatic process of extraction of **Indicators of Compromise (IOC)** that is daily run on dozens of new malwares, intercepted in the wide for populating our Knowledge Base.

Corrado Aaron Visaggio

Group Chief Scientist Officer & Malware Lab Director

a.visaggio@defencetech.it

1. Executive Summary

This document presents the open-source release of yara-x dotnet, a .NET wrapper library developed by our Malware Lab that provides C# developers with access to the yara-x binary pattern matching engine.

The library is released under the MIT license and distributed via NuGet as the DefenceTechSecurity.Yarax package.

yara-x dotnet bridges a critical gap in the .NET ecosystem by offering both high-level bindings that mirror the simplicity of the official Python API and low-level bindings that expose the full capabilities of the underlying C API, including proper multithreaded scanning support.

2. Background

2.1 YARA and Binary Signature Matching

For many years, YARA has been the de facto standard for writing binary signatures, declarative rules that identify files based on their content. It is extensively used across the cybersecurity industry for malware detection, threat hunting, incident response, and behavioral analysis of executable files.

2.2 yara-x: The Next Generation

yara-x is a complete rewrite of the original YARA engine in Rust, developed by the YARA project maintainers. It delivers improved performance, enhanced safety guarantees through Rust's memory model, and introduces new features while maintaining backward compatibility with existing YARA rules. yara-x exposes its functionality through a command-line interface and a C API for third-party integration.

While yara-x provides official bindings for Python, Go, and C, no official .NET bindings exist. This gap prevents the large community of C# and .NET developers, particularly in enterprise security tooling and Windows-centric environments, from leveraging the new engine directly. The purpose of yara-x dotnet is to address this gap.

3. Project Overview

3.1 Design Goals

- The library was designed with the following objectives:
- Provide idiomatic C# access to yara-x capabilities without requiring manual P/Invoke or unsafe code.
- Offer a high-level API that mirrors the simplicity of the official Python bindings for common use cases.
- Expose low-level bindings for advanced scenarios requiring fine-grained control, custom memory management, or multithreaded scanning
- Distribute precompiled native binaries for all major platforms via NuGet, eliminating the need for users to compile yara-x from source.
- Maintain full compatibility with the latest yara-x release.

3.2 Supported Platforms

The NuGet package includes precompiled native binaries for the following platforms:

Platform	Architecture	Runtime Identifier
Windows	x64	win-x64
Linux	x64, ARM64	linux-x64, linux-arm64
macOS	x64, ARM64	osx-x64, osx-arm64

These binaries are built from the latest yara-x release and include platforms for which the upstream project does not provide precompiled artifacts.

4. Installation and Usage

4.1 Installation

Install the NuGet package using the .NET CLI:

```
dotnet add package DefenceTechSecurity.Yarax
```

4.2 Basic Usage

The following example demonstrates compiling a YARA rule and scanning a file:
using DefenceTechSecurity.Yarax;

```
using var yrx = Yarax.Compile("""
    rule ExampleRule {
        strings:
            $a = "example string"
        condition:
            $a
    }
""");

var results = yrx.Scan(File.ReadAllBytes("sample.bin"));
```

For more examples and documentation, please refer to the GitHub repository.

5. Licensing and Distribution

yara-x dotnet is released under the MIT license, permitting unrestricted use, modification, and redistribution in both open-source and proprietary projects. The source code is hosted on GitHub, and the compiled package is available on NuGet.org under the identifier DefenceTechSecurity.Yarax.

This release reflects Defence Tech's commitment to contributing to the open-source cybersecurity ecosystem and supporting the broader community of security tool developers.

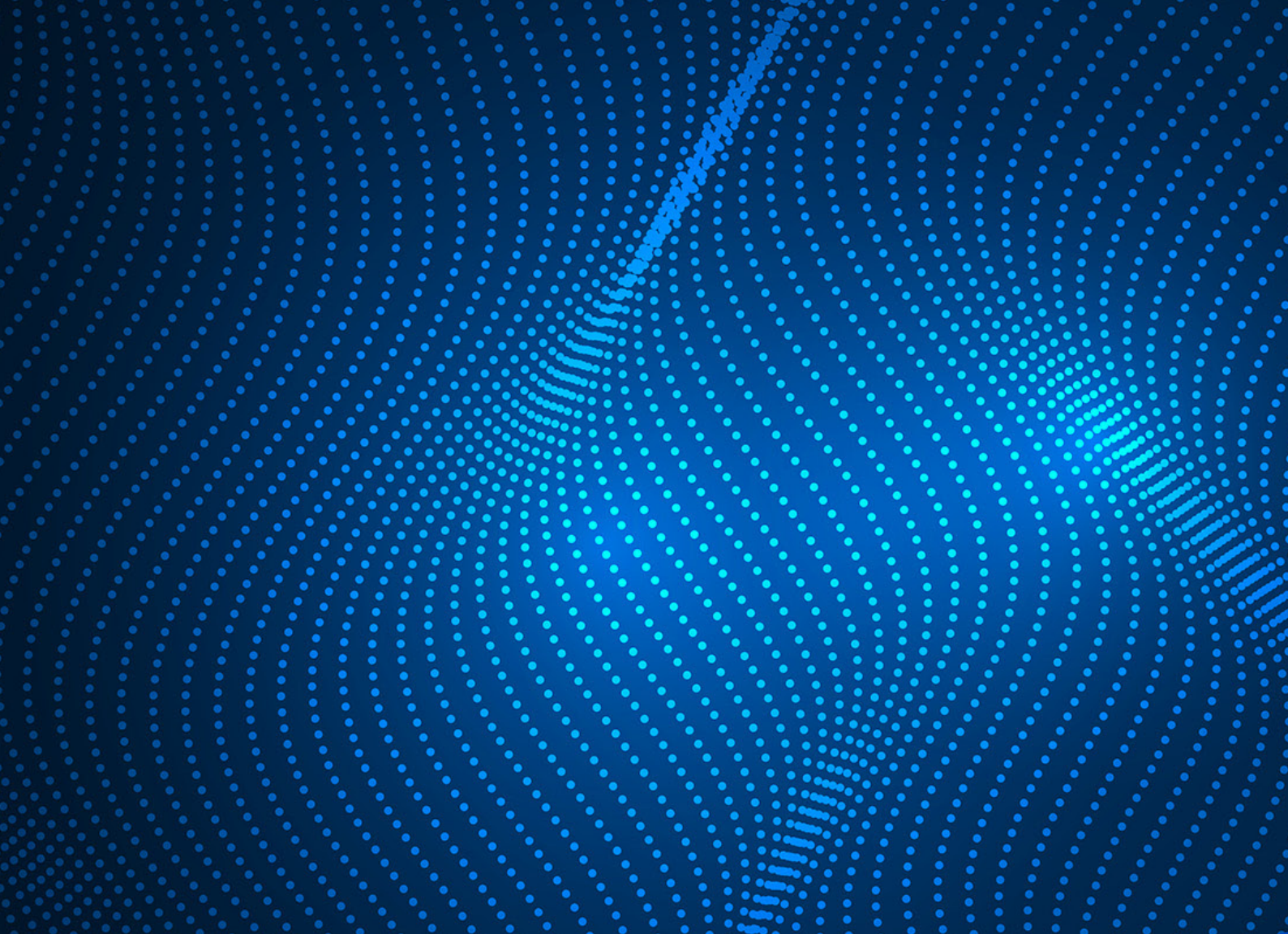
6. References

- yara-x project
https://github.com/VirusTotal/yara-x

- yara-x dotnet GitHub repository:
https://github.com/DefenceTechSecurity/yara-x-dotnet

- NuGet package:
https://www.nuget.org/packages/DefenceTechSecurity.Yarax

- YARA documentation:
https://yara.readthedocs.io



tinexta
defence

Next | Donexit | Foramil | Innodesi

Via Giacomo Peroni, 452 – 00131 Roma
tel. 06.45752720 – info@defencetech.it
www.tinextadefence.it

#TinextaDefenceBusiness