Malcious code detection tool

Static and dynamic analysis tool

Our software can do both static and dynamic detection analysis
1. static - using regex for pattern matching with locally saved malicious database without executing the code
2. Dynamic - by using docker creating an isolated environment run user's code for security and also to see how the code will react to sensitive data.

Tools to use

Qt: Cross-platform framework for developing graphical user interfaces (GUIs).

Regex Tool: A tool used for pattern matching or detecting specific text patterns.

Docker Environment: A platform to create isolated and portable containerized environments.

OpenSSL: A toolkit for implementing secure communication using encryption protocols.

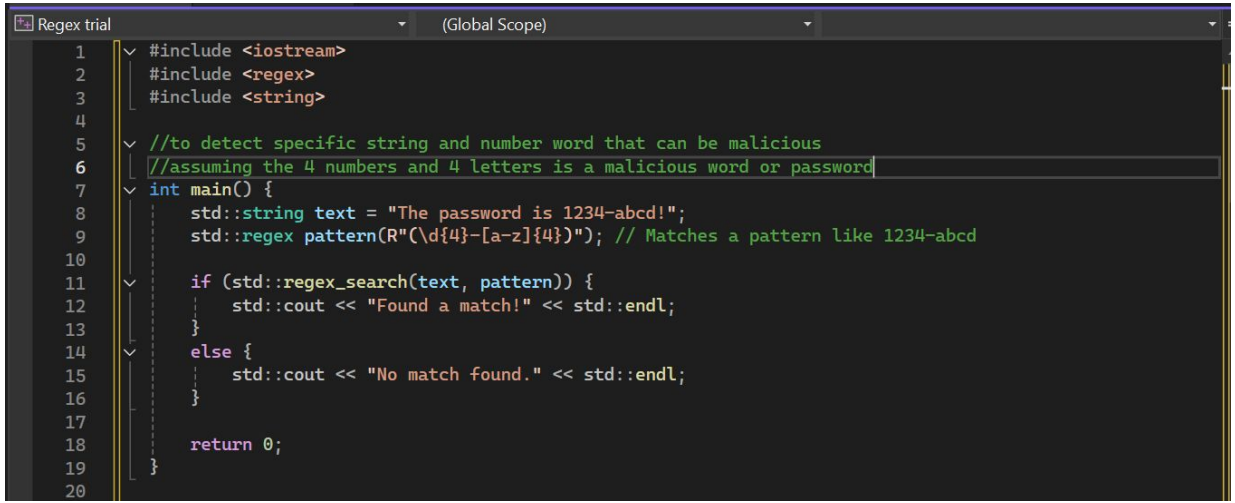
Inno Setup: A free installer tool for packaging software into a single executable file

Regex rules

```
//regex pattern

/*Character Classes:
¦
[a-z]: Matches any lowercase letter.
[A-Z]: Matches any uppercase letter.
[0-9]: Matches any digit.
Quantifiers:
¦
*: Matches zero or more occurrences (e.g., a* matches a, aa, or nothing).
+: Matches one or more occurrences (e.g., a+ matches a or aa).
{n}: Matches exactly n occurrences (e.g., \d{3} matches exactly three digits).
Anchors:
¦
^: Matches the start of a string.
$: Matches the end of a string.
Groups and Alternation:
¦
(...): Groups patterns (e.g., (abc|def) matches either abc or def).
Special Characters:
¦
\d: Matches any digit (equivalent to [0-9]).
\w: Matches any word character (alphanumeric + underscore).
\s: Matches any whitespace character (space, tab, newline).*/
/*
```

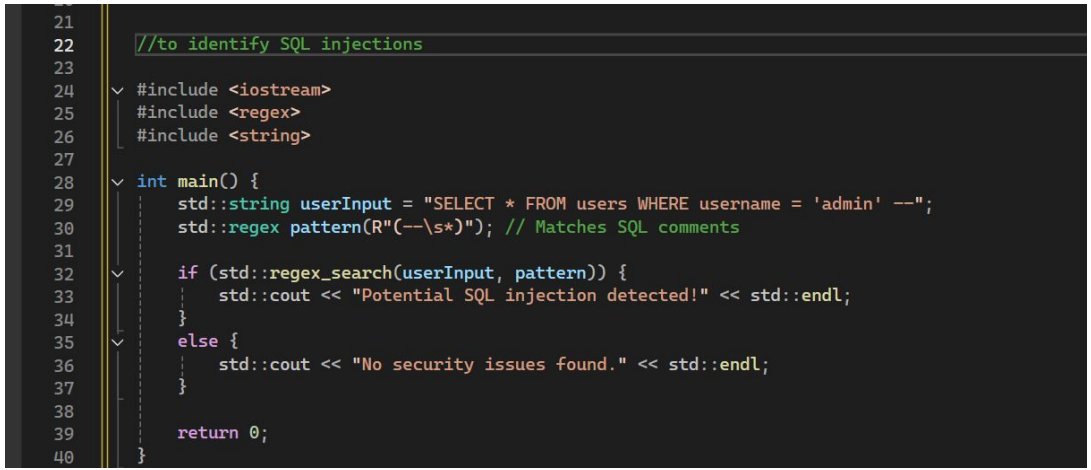
Regex Pattern matching sample code



Regex sample on how to detect sql

`--` is used in SQL injection which comments out the rest of the query, allowing attackers to bypass certain conditions or controls

An attacker could potentially use `//` in C++ code to comment out critical parts if they have access to the code. This could disable important logic or security checks, similar to how `--` is used in SQL injections



```cpp
//to identify SQL injections

#include <iostream>
#include <regex>
#include <string>

int main() {
    std::string userInput = "SELECT * FROM users WHERE username = 'admin' --";
    std::regex pattern(R"(--\s*)"); // Matches SQL comments

    if (std::regex_search(userInput, pattern)) {
        std::cout << "Potential SQL injection detected!" << std::endl;
    }
    else {
        std::cout << "No security issues found." << std::endl;
    }

    return 0;
}
```