5 Security Code Review

5.1 Czym jest code review?

Code review to inaczej przegląd lub inspekcja kodu. Polega na analizie kodu napisanego przez innego programistę po to, by wykryć potencjalne błędy. Po zbadaniu kodu źródłowego recenzent przekazuje autorowi swoje uwagi na jego temat – zarówno negatywne, jak i pozytywne.

Źródło: *https://www.holdapp.com/pl/blog/code-review-co-to-jest*

5.2 Czym jest security code review?

Security code review to ręczny lub zautomatyzowany proces, który bada kod źródłowy aplikacji. Celem tego badania jest zidentyfikowanie wszelkich istniejących błędów bezpieczeństwa lub luk w zabezpieczeniach. Przegląd kodu w szczególności szuka błędów logicznych, bada implementację specyfikacji i sprawdza między innymi wytyczne dotyczące stylu.

Zautomatyzowany security code review to proces, w którym narzędzie automatycznie przegląda kod źródłowy aplikacji, używając predefiniowanego zestawu reguł do wyszukiwania podatnego kodu. Zautomatyzowany security code review może szybciej wykryć błędy w kodzie źródłowym niż identyfikując je ręcznie.

Ręczna weryfikacja kodu polega na sprawdzaniu kodu źródłowego przez człowieka, linijka po linijce, w celu wykrycia luk w zabezpieczeniach. Ręczny security code review pomaga wyjaśnić kontekst decyzji dotyczących kodowania. Zautomatyzowane narzędzia są szybsze, ale nie są w stanie uwzględnić intencji dewelopera i ogólnej logiki biznesowej.

Źródło: *https://medium.com/@paul_io/security-code-review-101-a3c593dc6854*

Co (między innymi) powinniśmy sprawdzać podczas security code review?

- Czy dane wejściowe są odpowiednio walidowane i zabezpieczone przed atakami XSS, SQL injection itp.
- Czy użyte mechanizmy sesji są bezpieczne, czy zarządzanie sesją jest bezpieczne, a informacje o sesji nie są łatwo dostępne lub manipulowalne
- Czy dane są przesyłane za pomocą bezpiecznego protokołu
- Czy kod jest odporny na ataki logiczne, takie jak próby oszustwa w procesach biznesowych
- Czy obsługa błędów nie ujawnia poufnych informacji i czy błędy są logowane bezpiecznie
- Czy istnieją zabezpieczenia przed atakami typu brute force czy atakami na mechanizmy uwierzytelniania
- Czy operacje na plikach są odpowiednio zabezpieczone, aby uniknąć ataków typu Path Traversal czy Remote File Inclusion
- Czy są stosowane zabezpieczenia przed atakami na bazę danych, takie jak filtracja danych wejściowych i stosowanie przygotowanych zapytań
- Czy manipulacja linkami i ścieżkami jest odpowiednio zabezpieczona, aby uniknąć ataków typu Path Traversal czy Open Redirect
- Upewnić się, że hasła NIE są trzymane w kodzie źródłowym
- Zapewnić, że wszystkie używane biblioteki i zależności są aktualne, aby uniknąć luk w zabezpieczeniach
- Zapewnić, że w kodzie nie ma poufnych informacji, takich jak klucze API czy poufne dane, które mogą zostać wycieknięte
- ...

Linki

- https://medium.com/@paul_io/security-code-review-101-a3c593dc6854
- https://docs.python.org/3/library/pickle.html
- https://sekurak.pl/java-vs-deserializacja-niezaufanych-danych-i-zdalne-wykonanie-kodu-czesc-i/
- https://security.szurek.pl/owasp-8/
- https://codenga.pl/artykuly/poradniki/co_to_jest_serializacja

Zadania

Dla każdego zadania przygotuj raport, w którym wskażesz znalezione podatności. Dla każdej podatności określ CVSS. Sklasyfikuj podatność na podstawie OWASP Top 10. Dla każdej podatności zaproponuj sposób, w jaki można ją wyeliminować (możesz wskazać linijkę/linijki, które zawierają podatność i pokaż poprawioną, bezpieczną wersję kodu).

5.1 (a) Zapoznaj się z poniższym kodem zapisanym w języku Python. Co robi dany program?

(b) Wskaż linijki, w których znajduje się podatność / podatności

(c) Jak można wykorzystać te podatności?

(d) Jak można poprawić poniższy kod, aby wyeliminować podatność?

```
from http.server import BaseHTTPRequestHandler, HTTPServer
from http.cookies import SimpleCookie
import base64
import pickle

class MyServer(BaseHTTPRequestHandler):
    def do_GET(self):
        cookies = SimpleCookie(self.headers.get('Cookie'))
        if cookies.get('username'):
            username = pickle.loads(base64.b64decode(cookies.get('username').value))
        else:
            username = 'stranger'

        self.send_response(200)
        self.send_header("Content-type", "text/html")
        self.end_headers()
        self.wfile.write(bytes("<html><head><title>Hello</title></head>", "utf-8"))
        self.wfile.write(bytes("<body>", "utf-8"))
        self.wfile.write(bytes("<h1>Hello %s</h1>" % username, "utf-8"))
        self.wfile.write(bytes("</body></html>", "utf-8"))

if __name__ == "__main__":
    webServer = HTTPServer(('0.0.0.0', 5001), MyServer)

    try:
        webServer.serve_forever()
    except KeyboardInterrupt:
        pass

    webServer.server_close()
    print("Server stopped.")
```

5.2 (a) Zapoznaj się z poniższym kodem zapisanym w języku Java. Co robi dany program?
(b) Wskaż linijki, w których znajduje się podatność / podatności
(c) Jak można wykorzystać te podatności?
(d) Jak można poprawić poniższy kod, aby wyeliminować podatność?

```
import java.io.*;
import org.apache.commons.codec.binary.Base64;
import java.io.Serializable;

public class Secure {

    static class User implements Serializable {
        String username;
        String email;

        public User(String u, String e) {
            username = u;
            email = e;
        }
    }

    public static User userFromString(String str) {
        try {
            ByteArrayInputStream in = new ByteArrayInputStream(
                (new Base64(true)).decodeBase64(str));
            ObjectInputStream iin = new ObjectInputStream(in);
            User user = (User) iin.readObject();
            iin.close();
            in.close();
            return user;
        } catch (Exception e) {
            e.printStackTrace();
            return null;
        }
    }

    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

5.3 (a) Zapoznaj się z poniższym kodem zapisanym w języku NodeJS. Co robi dany program?

(b) Wskaż linijki, w których znajduje się podatność / podatności

(c) Jak można wykorzystać te podatności?

(d) Jak można poprawić poniższy kod, aby wyeliminować podatność?

```
const express = require('express')
const PID = process.pid;
const crypto = require('crypto');

function log(msg) {
        console.log(`[${PID}]` ,new Date(), msg);
}

const app = express();

app.get('/apphealth', function apphealth(req, res){
        log("Initiated health checking");
        res.send("Server is working\n");
});

function generateRandomString(){
        return crypto.randomBytes(200).toString('hex');
}

app.get('/computehash', function computehash(req, res){
        if(typeof req.query.iter === 'undefinied'){
                var iter = 10;
        }else{
                var iter = req.query.iter;
        }
        const hash = crypto.createHash('sha512');
        for(let i = 1; i < iter; i++){
                hash.update(generateRandomString());
        }
        res.send(hash.digest('hex') + "\n");
});

app.get('/helloworld', function helloworld(req, res){
        param = req.query.val;
        if(param){
                safePram = encodeURIComponent(param)
                        .replace(/&#126;/g, '+')
            .replace(/%20/g, '+')
            .replace(/%3C/g, '&lt;')
            .replace(/%3E/g, '&gt;')
            .replace(/%3D/g,'=');
                res.send("Hello World. Here is a link to helloworld page :
                                <a href='/" + safePram + "'/>Click!</a>");
        }else{
                res.send('Provide a param');
        }
});

const PORT = process.env.PORT || 5003;
let server = app.listen(PORT, () => log('Server running on: ' + PORT));
```

5.4 (a) Zapoznaj się z poniższym kodem zapisanym w języku Go (GoLang). Co robi dany program?
(b) Wskaż linijki, w których znajduje się podatność / podatności
(c) Jak można wykorzystać te podatności?
(d) Jak można poprawić poniższy kod, aby wyeliminować podatność?

```go
package main

import (
	"archive/zip"
	"io"
	"os"
	"path/filepath"
)

func Unzip(src string, dest string) ([]string, error) {
	var filenames []string
	r, err := zip.OpenReader(src)
	if err != nil {
		return filenames, err
	}
	defer r.Close()

	for _, f := range r.File {
		fpath := filepath.Join(dest, f.Name)
		filenames = append(filenames, fpath)

		if f.FileInfo().IsDir() {
			os.MkdirAll(fpath, os.ModePerm)
			continue
		}

		outFile, err := os.OpenFile(fpath,
			os.O_WRONLY|os.O_CREATE|os.O_TRUNC, f.Mode())
		if err != nil {
			return filenames, err
		}

		rc, err := f.Open()

		if err != nil { return filenames, err }

		_, err = io.Copy(outFile, rc)
		outFile.Close()
		rc.Close()

		if err != nil { return filenames, err }
	}
	return filenames, nil
}

func main() {
	sourceZip := "./archive.zip"
	destinationDir := "./"

	filenames, err := Unzip(sourceZip, destinationDir)
	if err != nil {
		panic(err)
	}

	for _, filename := range filenames {
		println("Extracted:", filename)
	}
}
```

5.5 (a) Zapoznaj się z poniższym kodem zapisanym w języku Python. Co robi dany program?

(b) Wskaż linijki, w których znajduje się podatność / podatności

(c) Jak można wykorzystać te podatności?

(d) Jak można poprawić poniższy kod, aby wyeliminować podatność?

```
import socket
import ssl

hostname = 'google.com'
context = ssl.create_default_context()
context.check_hostname = False

with socket.create_connection((hostname, 443)) as sock:
    with context.wrap_socket(sock, server_hostname=hostname) as ssock:
        ssock.write("GET / HTTP/1.1\r\nHost: {}\r\n\r\n".format(hostname).encode('utf-8'))
        print(ssock.read())
```

5.6 (a) Zapoznaj się z poniższym kodem zapisanym w języku PHP. Co robi dany program?

(b) Wskaż linijki, w których znajduje się podatność / podatności

(c) Jak można wykorzystać te podatności?

(d) Jak można poprawić poniższy kod, aby wyeliminować podatność?

```
#!/usr/bin/php
<?php
function verify($jwt) {
    list($h64, $d64, $sign) = explode(".", $jwt);

    if (!empty($sign) && (verify_signature($h64 . "." . $d64, $sign))) {
        die("Invalid Signature");
    }

    $header = base64_decode($h64);
    $data = base64_decode($d64);

    return parse_json($data);
}

function verify_signature($data, $signature) {
    $secret_key = 'your_secret_key';

    $expected_signature = hash_hmac('sha256', $data, $secret_key);

    return hash_equals($expected_signature, $signature);
}

function parse_json($data) {
    return json_decode($data, true);
}

$jwt = "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.
SflKxwRJSMeKKF2QT4fwpMeJf36POk6yJV_adQssw5c";

$result = verify($jwt);
print_r($result);
?>
```

5.7 *BigBlueButton is a purpose-built virtual classroom that empowers teachers to teach and learners to learn. In version 2.6.10 BigBlueButton is vulnerable to CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting').*

(a) W kodzie źródłowym `BigBlueButton` można znaleźć wiele miejsc podatnych na atak XSS, między innymi w pliku (dostępnym również na listingu **5.7**, poniżej): `bigbluebutton-html5/imports/api/guest-users/server/methods/setGuestLobbyMessage.js`.

(b) Wskaż linijki, w których znajduje się podatność / podatności.

(c) Jak można wykorzystać te podatności?

(d) Jak można poprawić poniższy kod, aby wyeliminować podatność?

(e) Przeanalizuj commit: `https://github.com/bigbluebutton/bigbluebutton/commit/304bc`, aby sprawdzić, jak deweloperzy poradzili sobie z tą podatnością.

```
import { Meteor } from 'meteor/meteor';
import { check } from 'meteor/check';
import RedisPubSub from '/imports/startup/server/redis';
import Logger from '/imports/startup/server/logger';
import { extractCredentials } from '/imports/api/common/server/helpers';

const REDIS_CONFIG = Meteor.settings.private.redis;
const CHANNEL = REDIS_CONFIG.channels.toAkkaApps;
const EVENT_NAME = 'SetGuestLobbyMessageCmdMsg';

export default function setGuestLobbyMessage(message) {
  try {
    check(message, String);

    const { meetingId, requesterUserId } = extractCredentials(this.userId);

    check(meetingId, String);
    check(requesterUserId, String);

    const payload = { message };

    Logger.info(`User=${requesterUserId} set guest lobby message to ${message}`);

    RedisPubSub.publishUserMessage(CHANNEL, EVENT_NAME, meetingId, requesterUserId, payload);
  } catch (err) {
    Logger.error(`Exception while invoking method setGuestLobbyMessage ${err.stack}`);
  }
}
```