COLLEGE OF ENGINEERING

Electrical Engineering and Computer Science

SECURITY-BASED RESEARCH PROJECT

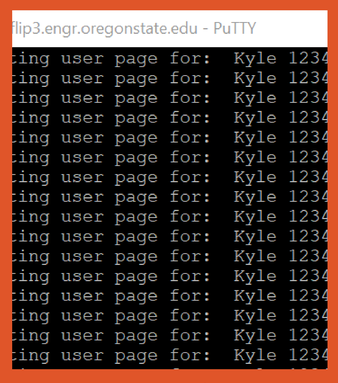
A pair of websites designed to demonstrate common security vulnerabilities, and how they might be addressed.

BACKGROUND

- The United States government estimates that malicious cyber actors cost the U.S. economy anywhere from $57-109 billion in the year 2016 alone (CEA, 2018).
- The Open Web Application Security Project (OWASP) Foundation created a list of the top ten cyber security risks, as well as security implementations to address the risks.
- The Secret Keeper project was designed and implemented at the request for an apparatus which illustrates the attacks outlined in the OWASP list. Three main components make up the project: a 'weak security' site, exploits designed to take advantage of the weak site's security flaws, and a 'strong security' site designed to stop the exploits.
- Both weak and strong sites allow users to create profiles to save private data which can later be accessed and modified.

TECH STACK

- Primary Languages: JavaScript (using Node.js and Handlebars), HTML, CSS, SQL
- Additional components: AJAX calls, Bash scripts

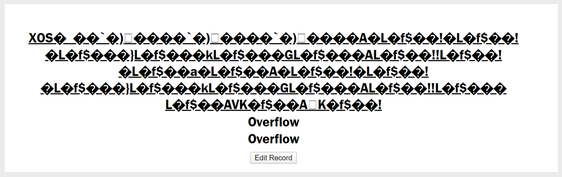


DoS Attack

THE 'NOT-SECURE' SITE

The "not secure" version of Secret Keeper allows the user to experience hacking in an ethical way. The README documents common exploits, with instructions on how to perform each attack. Attacks from the project include:

- Brute-force password attack (with both list-based and randomly-generated passwords)
- SQL injection attack
- HTML injection attack
- Buffer overflow attack
- Denial of service attack

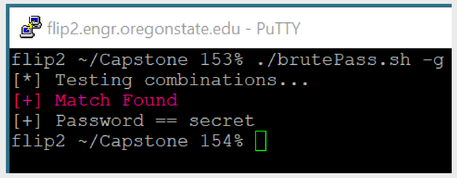


Buffer Overflow Attack

THE 'SECURE' SITE

The "secure" version demonstrates security features that address these vulnerabilities. These include:

- Encrypted data transmission
- Excess log-in-attempt delay
- Two-factor authentication
- SQL placeholders
- Variable-setting practices that disable script execution
- Input validation
- Hidden passwords at input



Brute Force Attack

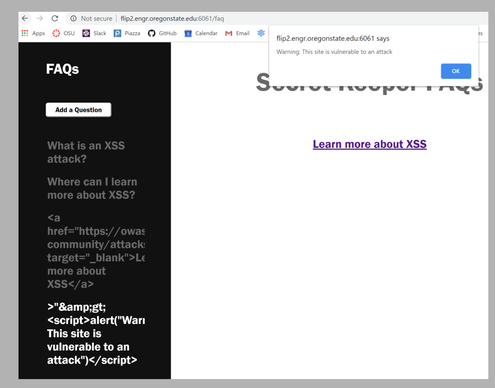
THE TEAM

Kyle Dixon | dixonky@oregonstate.edu

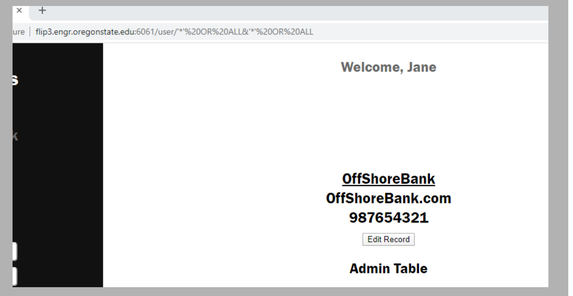
Katie Young | youngkat@oregonstate.edu


```
app.put('/',function(req,res,next){
    var uname = masks.removeMask([... req.body.user_name]);
    var upass = masks.removeMask([... req.body.user_pass]);
    var unameE;
    var upassE;

    mysql.pool.query("SELECT salt FROM salts WHERE user_name=?", [uname], function(error,
        if (error) {
            console.log(JSON.stringify(error));
            return;
        }
        var salt = results[0].salt;
        unameE = masksSalt.addMaskSalt([... uname],salt);   //Use salt to encrypt user cre
        upassE = masksSalt.addMaskSalt([... upass],salt);

        var useTokens = takeToken([uname].toString());      //Get token to initiate rate
        useTokens.then(() =>
            mysql.pool.query("SELECT COUNT(1) AS total FROM userE WHERE user_name=? and us
                if (error) {
                    console.log(JSON.stringify(error));
                    return;
                }
                const now = Date();
                console.log("Login attempt: ", uname," p: ", upass,  " @: ", now);
                res.send(results);
            })
        );
    })
});
```

Salt-based encryption and log-in attempt tracking reduces the amount of exploitable vulnerabilities.



HTML Injection Attack



SQL Injection Attack

Oregon State University