# SECCON 2019 - Beeeeeeeeeeer

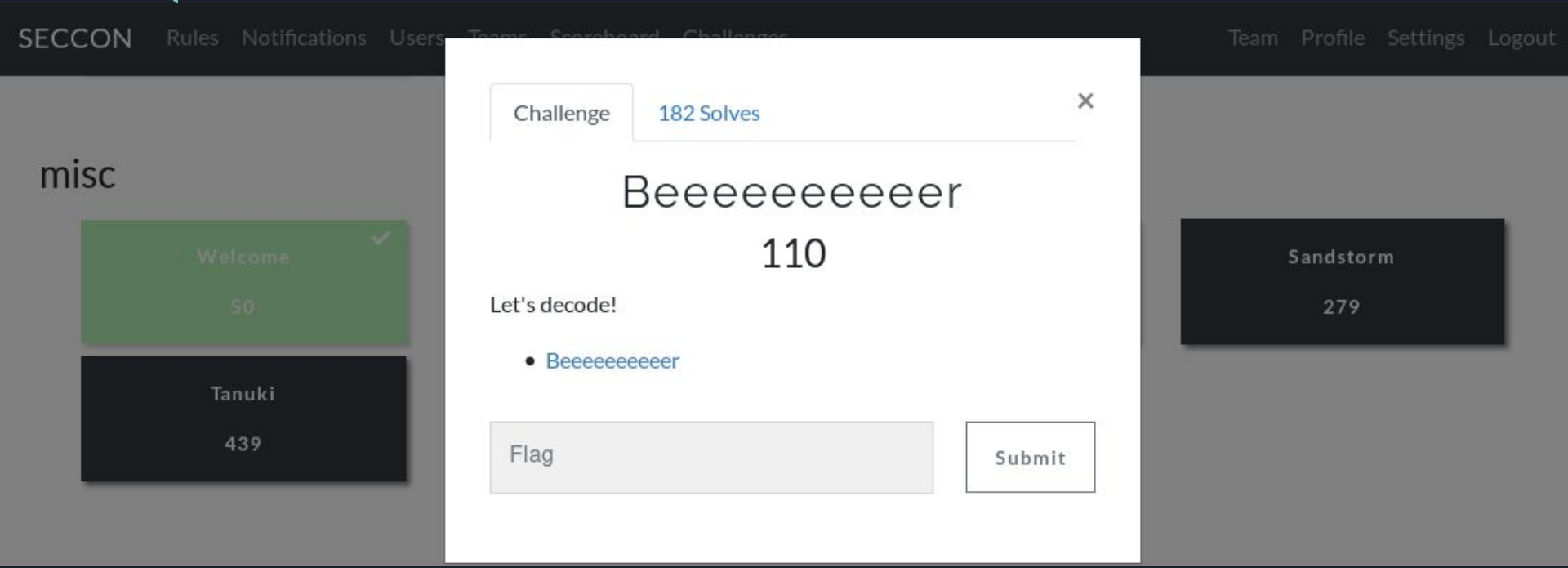
Felix Kehrer / @fkehrer

# Overview

Some recommendations for your talk:

- Provide an overview of the intended functionalities of the application
- Describe all the attempts you made to find the vulnerabilities, including unsuccessful ones (time permitting)
- Explain the exploitation steps in an understandable manner
- If possible, describe the impact of this security threat in a realistic scenario and discuss possible countermeasures

# Task


SECCON
Rules
Notifications
Users
Team
Profile
Settings
Logout
misc
Welcome
50
Tanuki
439
Sandstorm
279
Challenge
182 Solves
Beeeeeeeeeer
110
Let's decode!
Beeeeeeeeeer
Flag
Submit

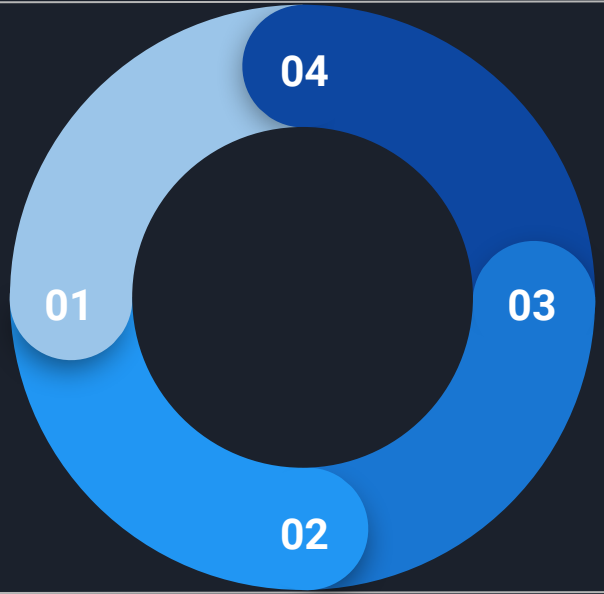
# Bash in 7 lines

| | |
|---|---|
| ./name | runs file *name* from local directory |
| ./A \| B | runs *A* and pipes its stdout to stdin of *B* (which also gets run) |
| ./A > B | runs *A* and writes the contents of stdout into file *B* |
| ./A && B | runs *A* and if it succeeds, runs B |
| $name | variable, holds a value |
| $(name) | gets evaluated to / replaced by the result of running *name* |
| : | historic relict, evaluates to true, used for comments |

# Algorithm

Run the script

Experience problems




Fix the problem or patch around it

Find the source of the problem

# Bash Debug?

Visual Studio | Marketplace

Visual Studio Code > Debuggers > Bash Debug

#!

## Bash Debug

rogalmic | 105,355 installs | ★★★★★ (4) | Free

A debugger extension for bash scripts (using bashdb).

**Installation**

Launch VS Code Quick Open (`Ctrl+P`), paste the following command, and press enter.

```
ext install rogalmic.bash-debug
```

Copy | More Info

## BASH Debugger

The Bash Debugger Project is a source-code debugger for bash that follows the gdb command syntax. The version 4.0 series is a complete rewrite of the previous series along the lines of my other POSIX shell debuggers and other debuggers mentioned below.

### Screenshots

Breakpoint

Backtrace

DDD

My latest project is a gdb-like Perl debugger, a rewrite of the classic Perl debugger. This debugger is also based on the slickest experimental debuggers for Rubinius, Ruby 1.9, and Ruby 1.8.

Also recent are debuggers for recent versions of Zsh and the Korn Shell, and a cool debugger for GNU Make.

# bash

> Bourne-Again SHell. `sh`-compatible command line interpreter. More information: https://gnu.org/software/bash.

- Start interactive shell:

```
bash
```

- Execute a command:

```
bash -c "{{command}}"
```

- Run commands from a file:

```
bash {{file.sh}}
```

- Run commands from a file, logging all commands executed to the terminal:

```
bash -x {{file.sh}}
```

- Run commands from a file, stopping at the first error:

```
bash -e {{file.sh}}
```

- Run commands from `stdin`:

```
bash -s
```

- Print the version information of bash (use `echo $BASH_VERSION` to show just the version string):

```
bash --version
```

# Overview

Some recommendations for your talk:

- Provide an overview of the intended functionalities of the application
- Describe all the attempts you made to find the vulnerabilities, including unsuccessful ones (time permitting)
- Explain the exploitation steps in an understandable manner
- If possible, describe the impact of this security threat in a realistic scenario and discuss possible countermeasures