Using Captcha Style Challenges to Defend Against BadUSB Attacks

Alexander Bladow Jensen Miller Mitchell White

# ABSTRACT

In modern computing there are plenty of security measures that are in place. For data, there is encryption, for your network there are firewalls, and for your hardware there are options such as TPM (Trusted Platform Module). These all are handled in the background for most users so they have the piece of mind that they will remain safe if they take a risky click or wind up on the wrong website. However, one long time security flaw is with devices that we plug in to our computer.

This flaw is that HID devices are innately trusted devices by a computer. Leveraging this innate trust, attackers were able to craft devices that pretend to be HID devices and deliver preprogramed payloads, often by autonomous typing. To address this security flaw, we have developed a tool that automatically detects device changes and actively attempts to defeat common USB security threats.

The solution we have crafted named BUCS (BadUSB Challenge System) is a software challenge that requires a live user to type in a random collection of dictionary words to prove that the device being plugged in is a legitimate HID device. To support our program, we have also created a set of test BadUSBs to test against.

# 1 INTRODUCTION

* Discuss history of USB attacks a little bit
* Explain problem statement
  + Top level why USB has issues
  + How these issues have been leveraged
* Other people’s attempts to fix the issue
  + Context and Provenance
  + Hardware solutions
* Our proposed solution
* Discuss the rest of the paper

# 2 BACKGROUND

* Explain USB security and why the vulnerability exists
  + Innate trust
* Explain the Scope of what we are doing
  + Software based solution vs autonomous typing
* Explain the beginnings of our defense mechanism
  + Catch the input before it can be placed maliciously
* Explain CAPTCHA

# 3 PROPOSED SOLUTION

* Present BUCS
  + Full explanation on what our program does and how it interjects to disrupt BadUSBs
* Discuss CAPTHCA Systems
  + Discuss the efficacy of CAPTCHAs
* Discuss implementation
  + Once we know more about how were making it
    - Explain event driven nature of program
    - Explain how it interjects
* Discuss Windows environment
  + What language and detection we are using, as well as what systems it works on
* Discuss Linux environment
  + Same stuff for windows but linux

# 4 BADUSB ATTACKS AGAINST MACHINES

* Discuss threat model
  + Hot plug attacks
    - Plug and play threats
    - Can do most anything
* Discuss the threats we have made
  + Show off our Bad USBs
    - Show timing of attacks
    - Show different types of attacks
      * Data exfil
      * Reverse shells
* Discuss the usage of BadUSB
  + Who will use it, why
  + Accidental vs intentional
* Cat and mouse game
  + Explain our attempts to defend and break our defenses

# 5 RELATED WORK

* Discuss other works found
* Hardware related solutions
* Software related solutions

# 6 AN OPEN ENDED PROBLEM AND DISCUSSION

* Once more testing is done, discuss where improvements lie
* Also where some tools still defeat
* Discuss advantages and disadvantages
  + Adv: no hardware install, easy to use, quick to install
  + Dis: Harder to achieve, wont defend against every type of BadUSB/plugin device

# REFERENCES