March 2020

INTEGRITY CHECK FOR USB DEVICES

HP INC

Integrity check for USB devices

This disclosure relates to the field of recognizing a security attack by a USB device on a computer. This method involves recognizing when a corrupted device would surreptitiously reconfigure itself and attack the computer system that is attached to.

The universal serial bus (USB) is a flexible interface for connecting all kinds of computer input and output device. Computer systems are generally very open and facilitate both physically accessible “walk up ports” to the exterior of the chassis and pre-install a very broad number of “class device” drivers that read the configuration and identification of the device and automatically enable the plugged-in device. USB is also commonly used as an internal connection to the computer, because of its low cost for implementation such devices such as touchscreens, fingerprint readers, memory card readers, etc. may function by the connection of USB to the device.

USB devices may present a security threat to the computer user. Some devices may physically appear to be memory disk on keys, or a USB Type-C power adapter, for example, but when connected assert themselves as a keyboard and use automated scripts to take control or corrupt the computer. These are popularly called “Rubber Duckies.” Because to the computer they device appears as simply a user added keyboard on a user accessible port, the computer may not distinguish the attached device as a hacking weapon and instead will assume that the data input is intentional user input. The computer may have no method of distinguishing a legitimate external keyboard from the Rubber Duckie hacking device.

Even trusted devices such as the ones inside the computer may be corrupted by a hacker so that they are re-tasked to operate as a different device. USB devices internal to a computer are often expected to never change, for example a specific physical port may be expected to be only a Bluetooth wireless module. But if the configuration of the device changes, software may be automatically reconfigured on the port and full operating privileges provided to it as if it were a walk-up port.

Methods of protecting against devices that masquerade as one device and then reconfigure as a chameleon (becoming a different device) have suggested Whitelisting, where the expected device on every host port is identified and a device presenting a different configuration is rejected. The white list is retested after an insertion, removal, or reset of the USB device. Internal ports are always compared to a “profile” and any change rejected. (Internal ports may have their known configuration recorded in system Bios).

What has not been distinguished in protection schemes before is the association between a link Reset and a physical removal of the device. A physical removal of an internal device is unexpected, as well an external device plugged into a walk-up port is not expected to change its Configuration without also a physical removal.

This idea discloses a method of detecting a dynamic alteration of the configuration of a USB device that suggests that an unauthorized, fake, doctored, or offending device has dynamically and surreptitiously altered its operation. The method could be used by a software agent operating on the host computer, for example the USB Bus Driver directly or an application that can examine the ports. The steps of operation;

1. The host software recognizes the number of USB ports available to the system, and distinguishes which ports are used for internal purposes, and which are available as external walkup ports. The distinction between internal ("Hidden"), and external ("Visible") ports is commonly provided by the ACPI (Advanced Configuration and Power Interface) _PLD method in a computer system.

2. A profile of the devices attached to each port is provided. The profile is an identity of the device, for example including the vendor ID (VID), product ID (PID), the class code and other configuration that is used by the host to activate the appropriate driver for the attached product.

3. The profile of the attached devices is changed at any time that the device is physically removed or attached to the port. Detection of the attached device may be accomplished in several ways, for example by direct detection of a USB Type-C device or by measuring the electrical characteristics of the data lines when the device is a USB Standard A port.

4. The host then waits to detect that a change in configuration has occurred on the port. Such changes may occur for example after a system reset, after a link reset, or initiated by the USB device.

5. Before enabling the USB device, the host performs checks;
   a. for the list of internal ports, the host compares that any change in the configuration of the detached device has occurred. Any change in configuration suggests a corrupted device is active.
   b. for the external ports, the host identifies any change in configuration that occurred without a physical removal and insertion cycle. Any change in configuration without a physical removal suggests a corrupted device.

6. Upon detection of a corrupted device, the host may alert the user or deny activation of the suspected device.

Disclosed by Lee Atkinson, Arthur Yang and Yu-Jen Yang, HP Inc.