Hello, everyone, I am here today to present our work, BadUSB-C: Revisiting BadUSB with Type—C. This is a course project of Computer Security instructed by Prof. Zhang in Southern University of Science and Technology.

Here is the outline of my today’s presentation. First, we will go through some background about USB protocol. Then, I will introduce the design of our BadUSB-C and a prototype we implemented to test its efficiency. After that, I will talk about a case study of a real-life scenario of sharing power bank, where we simulated stealing private data using BadUSB-C. After the case study, I will discuss the limitations, mitigations and the responsible disclosure we make to HUAWEI. Lastly, I will summarize and conclude this presentation.

USB, short for Universal Serial Bus. It is kind of a ubiquitous protocol nowadays. You can use it to connect HIDs like keyboard or mouse. You can use it to connect powerbank to charge your phone when going outside. You can also use it to carry data around with a flash drive.

What’s more, with Type-C connector introduced with USB 3.1. You no longer need to face a messy ball of cables.

All you need is a single USB Type-C cable.

But just as a saying goes, with great power comes great responsibility. There are all kinds of vulnerabilities were discovered during the development of USB protocol. In 1996, USB 1.0 was announced with support for basic HIDs like keyboard and mice. At that time, the BadUSB attack is already possible as there is few authentication mechanism in USB protocol. In 2013, USB 3.1 was released with a major change in the connector standard. In USB 3.1, a new connector standard called USB Type-C was introduced, which allows transmission of video data like DisplayPort over a USB cable. This also paved the path for our work, BadUSB-C.

As our work is an extension to the original BadUSB attack, now I will introduce the traditional BadUSB attack and its limitation. BadUSB attack is an attack presented on BlackHat 2014. It abuses the lack of authentication of USB protocol to trick the victim’s device into believing a flash drive is a keyboard, thus executing arbitrary malicious command. But it has a major drawback. As we can see here, after it executed a malicious command like a reverse shell. The attacker cannot obtain any feedback until the reverse shell is established, which largely limited this attack.

We summarize the drawback of traditional BadUSB attack as follows. First, due to the lack of a feedback channel during the attack process, the attacker cannot view the attack progress, thus cannot carry out precise attacks. Second, as the BadUSB only provides HID emulations, it is almost impossible for BadUSB to interact with GUI component like button. This severely limited the potential of BadUSB on mobile devices, which usually does not equip with a powerful command line. Apart from that, to establish a connection between the attacker and the victim’s device such as reverse shell. BadUSB typically require the host network usage which may be detectable to intrusion detection system or stopped by firewalls.

Here we introduce the design of our BadUSB-C and how we solve the three limitations of BadUSB. As we can see here, the BadUSB-C is connected to the victim’s device using a USB Type-C cable, which can transmit both video signal and HID signal.

In this way, our BadUSB-C can collect the video stream of victim’s device and build a stable feedback channel using the video capture card. This forms the video streaming path of our BadUSB-C and solves the first limitation of original BadUSB

What’s more, with this video feedback channel, the HID emulator here now can emulate not only keyboard, but also mouse to perform GUI interaction on the victim’s device. This solves the second limitation of original BadUSB.

Lastly, as we can see in this figure, our BadUSB-C uses a individual Wi-Fi/GSM module to transmitting data between the attack and the victim’s device. This avoids being detected by the firewall or IDS on the victim’s devices.

Here is a prototype we build to test the efficiency of BadUSB-C. A stands for the victim’s device. B is the compact version of our BadUSB-C after it is assembled. On the right is a decomposition of our BadUSB-C. We select a chip from ATMEG as the HID emulator and the Wifi chip and used a Raspberry Pi as the on-board computer to preprocess the video data to avoid transmitting too much data to the attacker. One difference between our prototype and our design is the auxiliary power bank, which provides power for both BadUSB-C itself and the victim’s device.

We also conduct a case study of the sharing power bank, which is a popular service in China. When you are outside and your phone is nearly depleted. You can rent a temporary power bank to charge your phone and return it later.

This is a typical attack procedure of BadUSB-C on sharing power bank. First, the attacker rents a power bank and replaces its internal with BadUSB-C.

Then the attacker returns the tampered power bank to the rental station.

After another user borrows the modified power bank and connect it to his/her own device.

The attacker has complete control over the victim’s device.

We performed an experiment on a HUAWEI P30 Android smartphone. Eleven applications were selected and tested in the following steps.

* We login a test account.
* Keeps all default settings.
* Attach BadUSB-C to the tested device.
* Simulate victim’s daily usage of the application and see what we got.

Here is some screenshot of the leaked informations.

And here is a table summarizing the leaked information during our experiment. In social network applications like WeChat and WhatsApp, the attacker can obtain the Contacts and Chat History of the victims. In Financial Applications like Alipay and Paypal, the financial status of the victim is leaked.

During our experiment, we also find that the BadUSB also has the following limitations. First, BadUSB-C cannot bypass biometric authentications like fingerprint. Second, BadUSB-C requires DisplayPort over USB Type-C feature to work, which currently only well supported on flagship phones. Lastly, BadUSB-C may incur notifications on victim’s devices and be discovered.

To mitigate this vulnerability, we also proposed a novel mitigation for this kind of problem. That is to divide UI rendering into two layers, one is the sensitive layer and the other is insensitive layer. As this figure suggested, sensitive contents like password are only rendered on sensitive layer and will not be transmitted to external devices. Thus, this mechanism effectively mitigated this issue.

After this vulnerability was discovered, we immediately contacted HUAWEI, who later assigned a CVE entry for this vulnerability. This is the email of HUAWEI’s response.

We also applied for the bug bounty program of HUAWEI and gained a reward over 4500$.

Now mitigation for this vulnerability has been deployed, which requires user authentication before allowing external USB devices.

We summarize our work as follows.

Firstly, we explore a new attack scheme leveraging the latest feature of USB protocol.

Secondly, we conducted a real-life scenario study of sharing power bank to test BadUSB-C efficiency.

Lastly, we propose a novel mitigation named Isolated UI Rendering for our BadUSB-C attack.