Summary

The Citizen Lab is an academic research group based at the Munk School of Global Affairs & Public Policy at the University of Toronto in Toronto, Canada.

We analyzed the version of Baidu Input Method included with a Samsung SM-T220 tablet as part of our ongoing work analyzing popular mobile and desktop apps for security and privacy issues. We found that this app includes a vulnerability which allows network eavesdroppers to decrypt network transmissions. This means third parties can obtain sensitive personal information including what users have typed. To address these issues, we suggest using HTTPS or TLS rather than custom-designed network protocols to encrypt sensitive network communications.

File name	Package Name	Version analyzed
BaiduInput.apk	com.baidu.input	8.5.20.4

Table 1: The version of Baidu Input Method that we analyzed.

Findings

We found that the version of Baidu Input Method that we analyzed transmitted keystroke information via UDP packets to udpolimenew.baidu.com. In the remainder of this section we explain how a network eavesdropper can decrypt the contents of these messages.

The version of Baidu Input Method that we analyzed encrypts keystrokes using a modified version of AES. When encrypting, the Baidu algorithm's key expansion is like that of standard AES, except, on each but the first subkey, the order of the subkey's bytes are additionally permuted. Furthermore, on the encryption of each block, the bytes of the block are additionally permuted in two locations, once near the beginning of the block's encryption immediately after the block has been XOR'd by the first subkey and again near the end of the block's encryption immediately before S-box substitution. Aside from complicating our analysis, we are not aware of these modifications altering the security properties of AES, and we have developed an implementation of this algorithm to both encrypt and decrypt messages given a plaintext or ciphertext and a key.

The version of Baidu Input Method that we analyzed encrypts keystrokes by applying the above modified AES algorithm in electronic codebook (ECB) mode in the following manner. First, the app uses a hard-coded 128-bit key, k_f = "\xff\x9e\xd5H\x07Z\x10\xe4\xef\x06\xc7.\xa7\xa2\xf26", to encrypt another, generated, key, k_m. The encryption of k_m is stored in bytes 64 until 80 of each UDP packet's payload. The key k_m is then used to encrypt the remainder of a zlib-compressed

message payload, which is stored at byte 80 until the end of the UDP payload. We found that the encrypted payload included, in a binary format which we did not recognize, our typed keystrokes as well as the name of the application into which we were typing them (see Figure 1).

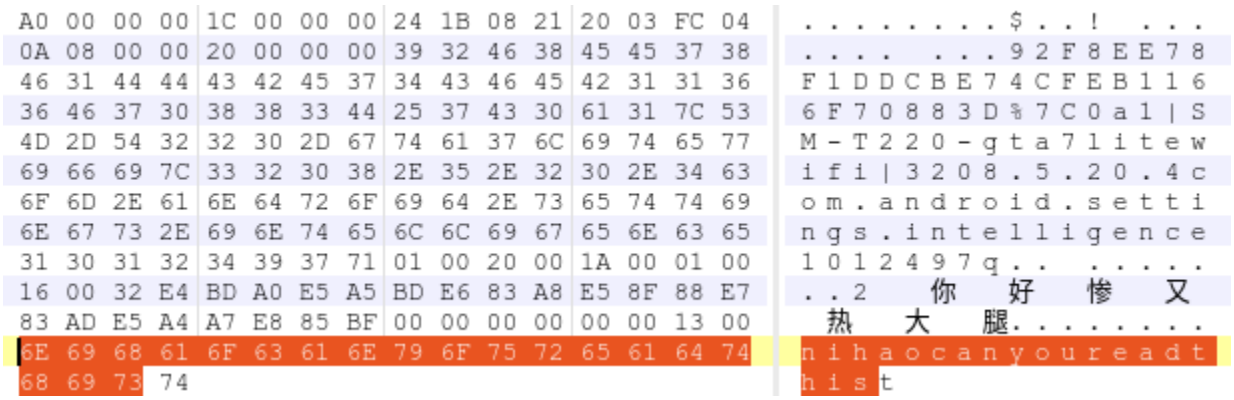

Figure 1: The decrypted and decompressed payload, revealing what we had typed ("nihaocanyoureadthis", highlighted) and the app into which it was typed ("com.android.settings.intelligence").

A vulnerability exists in the above protocol that allows a network eavesdropper to decrypt the contents of these messages. Since AES, including Baidu's modified AES, is a symmetric encryption algorithm, the same key used to encrypt a message can also be used to decrypt it. Since k_f is hard-coded, any network eavesdropper with knowledge of k_f can decrypt k_m and thus decrypt the plaintext contents of each message encrypted in the manner described above. As we found that users' keystrokes and the names of the applications they were using were sent in these messages, a network eavesdropper who is eavesdropping on a user's network traffic can observe what that user is typing and into which application they are typing it by exploiting this vulnerability.

Additionally, we found that key k_m was not securely generated using a secure pseudorandom number generator (secure PRNG). Instead, it was seeded using a custom-designed PRNG that we believe to have poor security properties, and, instead of using a high entropy seed, the PRNG generating k_m was seeded using the message plaintext. However, even without these weaknesses in the generation of k_m, the protocol is already completely insecure to network eavesdroppers as described in the above paragraphs.

Mitigation

In order to address the reported issues, Baidu Input Method should secure all transmissions using a popular, up-to-date implementation of HTTPS or, more generally, TLS instead of relying on custom-designed cryptography to secure the transmission of sensitive user data.