November 20, 2018

MULTI-DEVICE AUTHENTICATION FOR WINDOWS ENVIRONMENT

HP INC

Multi-Device Authentication for Windows Environment

Abstract: A secure technique for implementing multi-device-based authentication on a Windows operating system is provided using a threshold signature mechanism.

This disclosure relates to the field of computer security.

A technique is disclosed that provides multi-device based authentication for a Windows operating system.

The Windows operating system supports password, PIN, biometric, smartcard-based, and device-based authentication. However, all these have security and usability issues. The Windows operating system does not inherently support multi-device based authentication.

A prior solution allows a user to use their phone to lock and unlock their PC using Bluetooth proximity. When this feature is enabled, the PC automatically locks when the user walks away from the PC with their phone, and automatically unlocks when the user walks back to the PC with their phone. This mechanism requires the smartphone and the PC to be paired using Bluetooth. The device proximity is determined by using the Bluetooth received signal strength indicator (RSSI). The PC tracks this RSSI value. When the device moves away from the PC, this RSSI value drops. Once the RSSI value drops below a certain threshold, this indicates that the device has moved sufficiently far away from PC, and the PC will then lock after a few seconds. When the device moves closer to the PC, the RSSI value will increase. At a certain point, this value will be greater than a certain threshold. This indicates that the device has moved close enough to the PC, and the PC will unlock.

However, this technique has the same issue as any other authentication method based on a digital signature: if the secret key which is stored on a single device is stolen by an attacker then the attacker will be able to authenticate on behalf of the user. As a result, the technique is insecure.

According to the present disclosure, a multi-device based authentication protocol (MDAP) provides secure authentication to a Windows machine using multiple devices that are in a user's possession. Security is achieved by using a threshold signature mechanism.

A user owns a login credential and plural personal devices. These devices are ones which are closely associated with the user or some aspect of their identity such as, for example, a mobile phone, laptop, smart watch, or health monitor.

A verifier service establishes the identity of an authenticating client/user. It performs a threshold signature based authentication to verify the purported identity of an authenticating client. In one example, the verifier is a device running a Windows OS.

A challenger receives the challenge from the verifier and distributes it to the user's personal devices. The challenger receives information about the personal devices from the dealer, and establishes pairing with them to facilitate secure distribution of the data.

A combiner submits the signed challenge to the verifier on behalf of the client by combining the partial signature shares that it receives from the personal devices. It collects the partial shares from the user's personal devices, verifies these individual signatures, and then combines a threshold number of them to produce a group signature.

A dealer distributes credential shares to the personal devices. It establishes pairings with the user's personal devices to facilitate secure distribution of the data. In one example, the dealer is a smartphone.

The protocol has a Setup phase and an Authentication phase.

In the Setup phase, the dealer (smartphone) and the verifier (Windows PC) each generate a public-secret key pair by running Gen algorithm that create the keys. The dealer and verifier both send their public keys to each other. The user then securely pairs their personal devices with the dealer. The dealer then takes the group secret key and generates partial shares for each of the user's personal devices from it. Based on the input from the user, the dealer selects the total number of shares into which to split the group key, and the minimum threshold number of shares required for authentication. Once the shares are generated, the dealer then distributes the shares to each personal device over an authenticated channel and deletes the secret key. The dealer (smartphone) also acts as a combiner and a challenger in this scenario. The verifier asks the user to enter their account credentials and verifiers them against the user's windows login account. It then generates an AES verifier key and encrypts the credentials with it to generate encrypted credentials. By using the dealer's public key, the verifier encrypts the AES verifier with it to generate an AES dealer key.

In the Authentication phase, the dealer sends an authentication request to the verifier. The verifier performs a challenge by generating a fresh random message, signing it with the verifier's secret key, and sending it to the dealer along with the AES dealer key. The dealer receives the message and the AES dealer key, and verifies the signatures on the challenge message against the verifier's public key. If the signature verifies, the dealer sends the challenge to the user's personal devices using an authenticated channel. The personal devices receive the challenge message from the dealer (who acts as a challenger). The devices produce a partial signature by using their secret shares and send the partially signed message to the dealer (who acts as a combiner). The dealer verifies the correctness of the partial signatures received from each personal device, computes combined signatures, and verifies the combined signatures. If the signatures are valid, the dealer decrypts the AES dealer key with its secret key to get the AES verifier key. The dealer then encrypts the AES verifier key with the verifier's public key, concatenates the challenge message and the encrypted key, signs the response, and sends the response to the verifier. The verifier (Windows PC acting as an authentication service) verifies the response and, if verified, decrypts the encrypted AES verifier key and uses this key to decrypt the user's encrypted credentials. It then uses the decrypted credentials to log the user into her account.

Disclosed by Gurchetan Grewal and Josh Serratelli Schiffman, HP Inc.