September 20, 2016

ESTABLISHING PROXIMITY-BASED TRUST OF NEARBY DEVICES THROUGH AUDIO COMMUNICATION

David Aristizabal

Adam Liss

ESTABLISHING PROXIMITY-BASED TRUST OF NEARBY DEVICES THROUGH AUDIO COMMUNICATION

ABSTRACT

A method for establishing proximity-based trust of nearby devices through audio communication is disclosed. The method includes a protocol for devices that communicate over audio within a predetermined distance from each other to verify proximity. A packet token is included at the end of each packet. After each packet ends broadcasting, the sending device listens for a response from the receiving device for a window of time corresponding to the time it takes for sound to make a round-trip through the predetermined distance. This mechanism could be used in mobile application development where applications are required to ensure co-presence within a specified distance from another device as a criterion for trust.

BACKGROUND

Currently, direct audio communication between nearby mobile devices is possible. Typically, the devices exchange some kind of token over audio by sending an ultrasound message which helps in establishing co-presence. An audio communication could be intercepted from relatively long distances, for example, by using a parabolic microphone. So, an attacker could easily obtain co-presence tokens from a distance and trick other devices into thinking the attacker's device is nearby. By broadcasting at a high volume or by using an acoustic mirror, the attacker's device may also communicate with other devices at a distance. For example, this technology could be used to easily share a Wi-Fi key with guest devices or to seamlessly connect new Internet of Things devices to a user's home network, but only if both devices are within a predetermined distance. An attacker outside the user's home network can initiate a request and

by misrepresenting itself as a trusted entity, steal the Wi-Fi key, thereby gaining unauthorized access to the user's home network.

In another example, a driver may want to unlock a car with a smartphone application that operates via the cellular data network. However, this leaves the car vulnerable to hackers anywhere on the Internet. That vulnerability is eliminated by requiring the user to be physically near the car.

As a third example, many residential gas and electric meters can now be read electronically. It is conceivable that a criminal can determine when a homeowner is away for an extended time by "war driving" through a neighborhood and reading the meters on each house. This becomes impossible if the reader must be within several feet of the meter in order to read it.

To satisfy this requirement, a method that ensures co-presence of devices within a specified distance is proposed. The method verifies proximity of the intended device and thereby thwarts security attacks and privacy breaches.

DESCRIPTION

A method for establishing proximity-based trust of nearby devices through audio communication is disclosed. A protocol by which two devices could communicate over audio, ensuring that they are within a predetermined distance from each other, is presented. The protocol is implemented when a user wishes a device to initiate communication with another. The method is as follows:

1. The two peer devices (A and B, respectively) take turns to broadcast packets over audio.
2. After either device sends a packet, it listens for a response from the other device for a predetermined window of time. The window corresponds to the time it takes for sound to make a round-trip through the maximum allowable distance between A and B.

3. Each packet ends with a random packet token, ensuring the receiver must wait for the entire packet before it can initiate the response.
4. Upon receiving any packet, each device returns its token in some form, in addition to any data payload it transmits.
5. When a device receives a packet within the allowed window, it verifies that the previous packet token is valid, thereby ensuring proximity of its peer. Packets that begin outside the window, or that contain invalid tokens, are discarded as fraudulent.

An implementation of the method, as shown in FIG. 1, illustrates three devices: two user devices, A and B, and an attacker device, E. A predetermined distance D, measured from device A is designated as safe distance. Devices A and B are near each other at a distance less than D and are trusted to communicate with each other by virtue of their proximity. Device E represents a potential attacker's device, at a distance greater than D from both A and B. E is not trusted to communicate with A because it is not close enough.

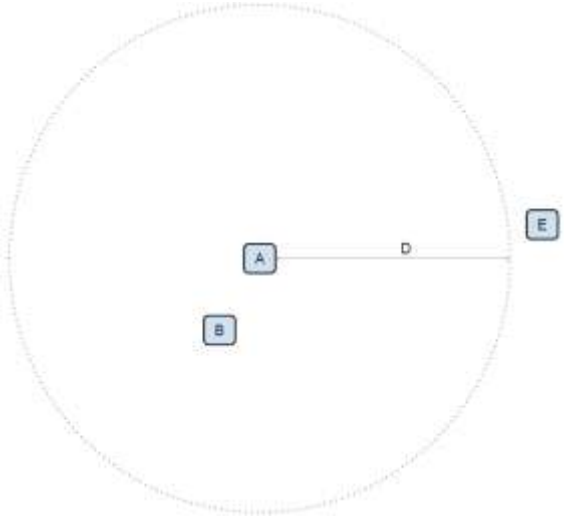

FIG. 1: Establishing co-presence of devices placed within A's sphere of trust, *i.e.* within a trusted distance from A.

It is possible for the attacker E located beyond the predetermined distance D to broadcast a packet ahead of time such that it arrives at one of the devices during the listening window. The

packet token mechanism described above ensures that such a packet from attacker E would be invalid, since the attacker would not be able to predict the value of the packet token ahead of time, *i.e*. before the sound arrives at the attacker's location. The time window described above is such that an attacker exactly at the predetermined distance would not have time to broadcast a packet after receiving the packet token, and have that packet arrive at A within the listening window.

Timing for a packet sent from device A and a response from B, as well as how the listening window of time physically prevents a valid packet sent by E from reaching A in time, are illustrated in FIG. 2. Note that device E could send a valid packet after receiving A's packet token, which is at the end of A's message. B sends a response packet, which arrives within A's listening window. Device E tries to send a valid packet from exactly distance D, but it arrives at A at the end of the listening window or beyond, so A ignores it, as shown in FIG. 2.

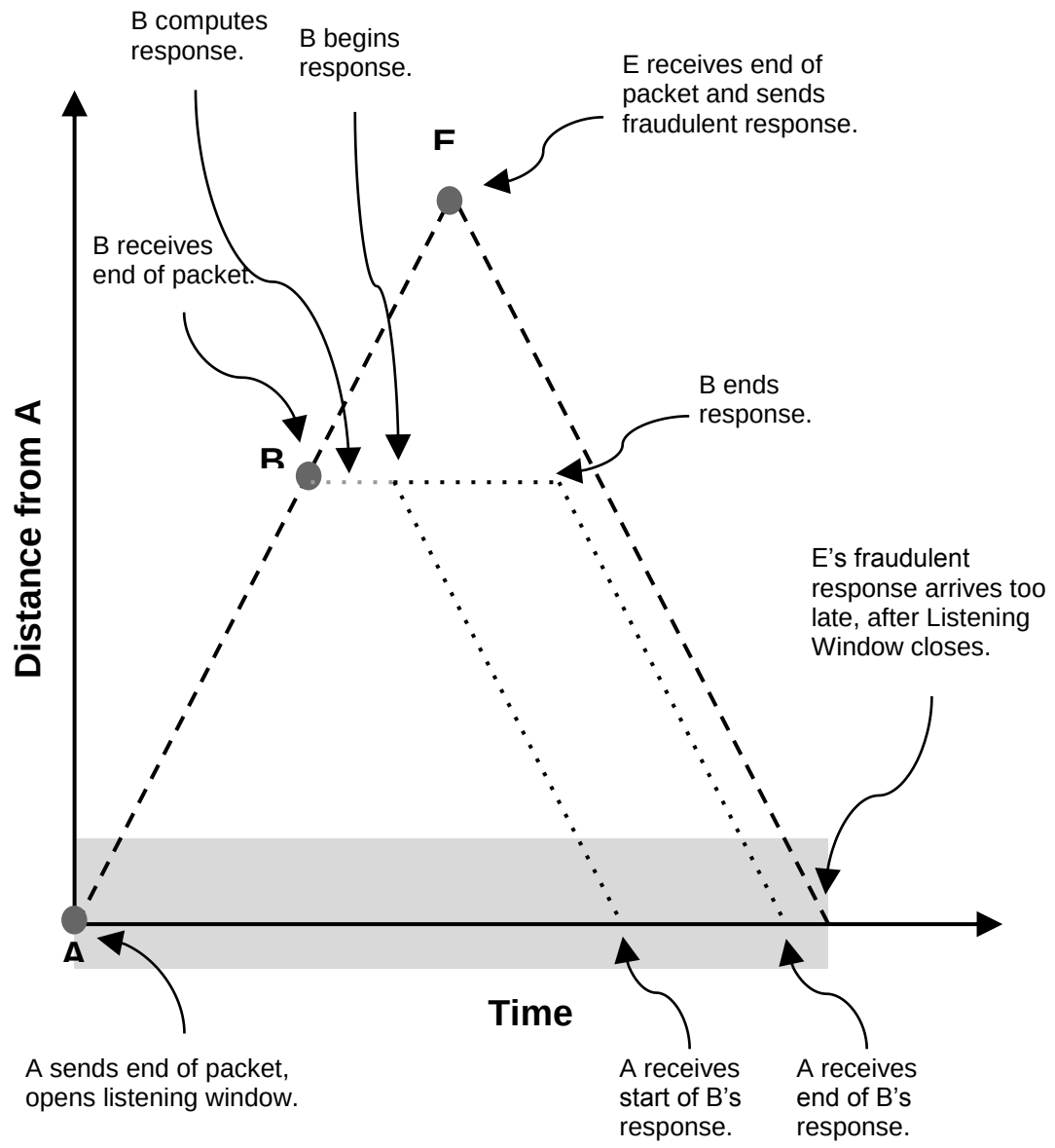

FIG. 2: Listening window of time for ensuring co-presence of devices

In the implementation of the method, the packet token is included at the end of each packet to ensure that an attacker must wait until the end of the packet arrives before it can craft a valid response packet. Each packet is cryptographically combined with the most recently received packet token in such a way that an attacker cannot craft a valid response packet without first receiving the entire packet token. The purpose is to ensure that every bit of the response

depends on every bit of the packet token. A simple implementation of this step might use the previous packet token as a seed to generate a random stream of bits and exclusive or (XOR) the result with the bits of the packet before transmission.

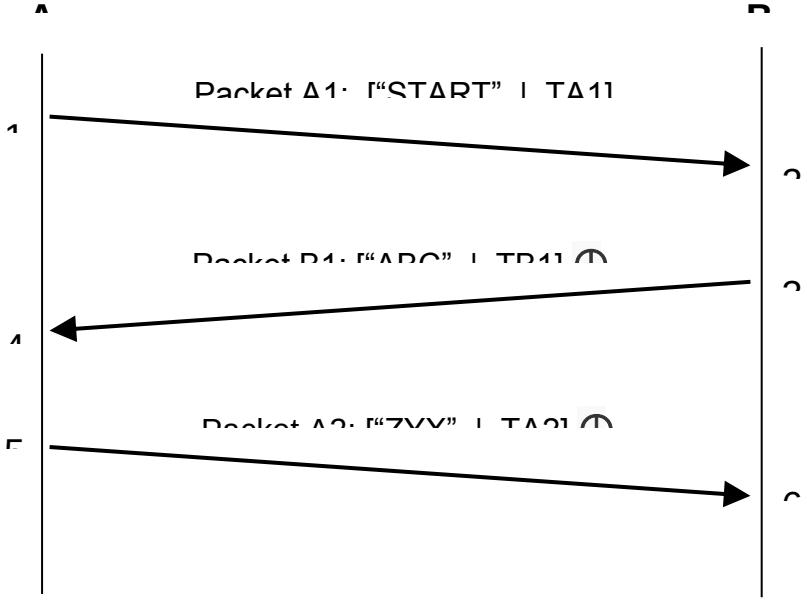

FIG. 3: Sequence of packet transmissions

FIG. 3: illustrates the following sequence:

1. A initiates communication and appends a random packet token, TA1.
2. B receives Packet A1.
3. B responds with "ABC" and appends a new random packet token, TB1. It uses TA1, the token it received from A, to seed a string of random bits, ***R***(TA1), which it XORs with its reply.
4. A receives the packet, computes the same string of random bits, ***R***(TA1), and again XORs it with the packet to restore the original data.
5. A responds with "ZYX" and appends a new random packet token, TA2. It uses

token TB1, which it received from B, to generate ***R***(TB1), which it XORs with its reply.

6. B receives the packet and XORs it with ***R***(TB1) to restore the original data.

It is not necessary that each packet contain an entire message, as the packets must arrive in order and can accumulate within each device until a complete message has been received. Each packet contains a relatively small number of bits, so long messages are split into multiple packets.

Standard mechanisms for error detection and correction may be used. However, a retry limit should be enforced to prevent an attacker from simply guessing the packet tokens until it succeeds.

Definitions and equations describing physical properties and limitations associated with the method are presented:

Description		**Units**	**Notes**
Predetermined maximum safe distance	D	m	
Distance between two nearby devices	d	m	d < D
Bit rate of data transmitted over audio	b	bits/s	
Speed of sound in air*	v	m/s	Worst case: v = 360 m/s
Listening window	Tw	s	Tw = 2D/v
Maximum length of response	Tt	s	2(D - d)/v
Maximum packet length	B	bits	B = floor(b × Tt) = floor[2b(D - d)/v]

* The speed of sound in air varies with several factors, of which temperature and humidity are the most significant. Its approximate range is from 330 m/s at 0° C and 0% humidity to 360 m/s at 40° C and 100% humidity. This information can be used to calculate the

listening window using the worst-case conditions (highest speed of sound, leading to the shortest listening window) for reasonable environmental conditions.

The maximum packet length depends greatly on how close the two devices are and the predetermined maximum safe distance. Examples using arbitrary but practical parameters are shown below:

Given D = 10 m, d = 5 m, b = 1000 bits/s
Tw = 55.6 m
Tt = 27.8 ms
B = 27 bits

Given D = 10 m, d = 2 m, b = 1000 bits/s
Tw = 55.6 m
Tt = 44.4 ms
B = 44 bits

The packet layout could be used from existing protocols for audio transmission, with slight modification to include the packet token at the end. If not modifying an existing protocol, here is an example of a possible way to lay out a packet.

Length	Payload	Checksum	Packet Token

Here, everything but the length is cryptographically combined with the previous packet token (except on the first packet).

Length: Size of the entire packet.

Payload: Message data.

Checksum: Used for error detection or correction.

Packet Token: Ensures proximity, per discussion above.

In some variations of the method, the listening window may be modified such that the condition is for the beginning of the packet to arrive within the window, and then the devices continue to listen until the end of the packet however long it may be, along with some kind checksum or hash mechanism to ensure that E is not able to modify the portion of the packet that arrives outside of the listening window.

In another variation, devices are allowed to transmit additional packets during their listening window to increase data throughput, in effect sending two consecutive packets and expecting two consecutive, corresponding responses.

In another variation, instead of assuming some worst-case distance between trusted devices, they could measure the response time to estimate their actual distance and adjust the maximum packet size accordingly. The closer the devices are together, the bigger the packets the devices could send within the listening window.

In another variation, the disclosed method could be extended to an arbitrary number of *n* devices. For example, device A could verify the proximity of the other devices, B, C, *etc.* in turn. Once this is completed, device A will have verified that all other devices are nearby, and all devices will also have verified that A is near them. C can then ask A to verify that B is nearby.

In an example application, devices could exchange public encryption keys using the disclosed protocol. After the exchange of public keys, the devices can use those keys as proof of initial proximity and continue to communicate over any mechanism using standard Public Key Encryption methods for signing and encryption. For example, device A would send a message to

B encrypted and signed in a way that ensures A knows only B can decrypt it, and B knows only A could have signed it.

In a second example application, one device, A, can securely share a secret, like a Wi-Fi network key, with a nearby device, B. The user would initiate the process, *e.g.* by pressing a button on each device. The devices would establish proof of proximity and exchange public keys as described in the example application above. Device A can then transmit the secret encrypted and signed such that only B can decrypt it and verify that it was sent by A. The transmission can happen over the disclosed method, or another mechanism, *e.g.* a different audio protocol, Bluetooth, Ethernet, Wi-Fi, etc.

In a third example application, two internet-connected devices, A and B, can establish proof of proximity prior to interacting with each other through a cloud service. The devices would establish proof of proximity and exchange public keys as described in the example above. A cloud service can then broker a high-bandwidth connection between the two devices, where each device uses standard mechanisms for securing their connection to the cloud service, e.g. HTTPS, SSL, TLS.

The described protocol does not gather, use or store personal information about users. In situations in which specific implementations using the described protocol may collect or use personal information about users (e.g., user data, information about a user's social network, user's location, user's biometric information, user's activities and demographic information), users are provided with one or more opportunities to control whether the personal information is collected, whether the personal information is stored, whether the personal information is used, and how the information is collected about the user, stored and used.