Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: May 17, 2018

S. Cheshire
Apple Inc.

November 13, 2017

Private Discovery Threat Considerations
draft-cheshire-dnssd-privacy-considerations-01

Abstract

This document provides a framework for evaluating and comparing solutions for privacy-respecting discovery mechanisms.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on May 17, 2018.

Copyright Notice

1. Introduction

When AppleTalk was introduced in 1986, privacy concerns were not foremost in most people's minds. The fact that a printer was offering printing service was not considered a secret, and the fact that a computer was seeking printing service was not considered a secret. The fact that the computer could discover the printer without expert configuration was considered remarkable.

Thirty years later, the landscape has changed. We now have many more network service types, and mobile wireless devices offering and consuming those services are common. Those mobile wireless devices and the services they offer or use often involve sensitive financial or medical data. Furthermore, the ubiquity of such mobile wireless devices makes them an attractive target for mischievous or outright criminal activity. The fact that a person's smartphone is communicating with their implanted glucose monitor or insulin pump is not something that should be public information.

Hence there is now a need for discovery mechanisms that utilize privacy-preserving techniques. There have been various different efforts to address this, but they tend to offer solutions based on assumptions of what privacy aspects are important, without articulating what those assumptions are. Without knowing the assumptions and design goals of a particular proposal it is hard to evaluate whether that proposal meets those goals, or indeed whether they are the right goals.

Without advocating for any particular solution, this document presents an overview of the various aspects of device discovery and service discovery, and outlines the privacy concerns of each. Any given proposal may not address all possible privacy concerns. Depending on the scenario, it may not be necessary to address every privacy concern. Indeed, it may turn out to be impossible, or at least impractical, to address all possible privacy concerns. This document provides a framework to help evaluate whether a given solution meets the privacy needs of some particular usage scenario.

2. Discovery Operations

Device discovery and service discovery involve three principal operations:

1. Offer
2. Discover
3. Use

The "Offer" operation is how a device offers a service on the network. Typically this involves, using today's terminology, (a) a "listening" UDP or TCP socket, which accepts incoming packets or connections, and (b) a way of advertising to other local and remote devices what kind of service is being offered, its name, and other metadata including how to reach it. Observe that there are three levels of information in use here: (i) the type of service, (ii) the name of the particular instance of that type of service, and (iii) the operational details of how to connect to and make use of that particular instance.

The "Discover" operation is how a client device learns what service instances are being offered (by local devices, and/or remote devices, depending on the discovery mechanism being used). Typically a client device knows what kind of service it is seeking, and wants to discover named instances of that service. The "Discover" operation is linking information level (i) type of service, with information level (ii) names of specific instances offering that type of service. The "Discover" operation can be viewed as providing a little information (just the name) about many different instances. In terms of complexity and efficiency, it's a 1 x n operation, getting one piece of information about n instances.

The "Use" operation is how a client device requests additional information (IP address(es), port number, and possibly other metadata), and then uses this information to communicate with the service instance and make use of the service it offers. The "Use" operation is linking information level (ii) specific instance name, with information level (iii) detailed information about that individual instance. The "Use" operation can be viewed as providing a lot of information about one particular instance. In terms of complexity and efficiency, it's an m x 1 operation, getting m pieces of information about 1 instance, and then proceeding to use that instance.

All three operations, and the three levels of information they use, need to be considered from a privacy perspective.

Note that some discovery mechanisms conflate "Discover" and "Use" into a single operation. Instead of requesting a little information about a lot of instances, or a lot of information about a single instance, they are only able to request everything about everything. They replace a 1 x n operation and an m x 1 operation with a combined m x n operation, always requesting m pieces of information each about n different instances.

3. Trust Granularity

When we talk about entities trusting other entities, what entities are we talking about?

Are the entities physical devices, like a smartphone or laptop computer?

Are the entities human users? If a device like a laptop computer has multiple users, we should not assume that because one user is authorized to discover certain services that means that all other users of that laptop are also authorized to discover those services.

Are the entities software applications? If a device like a smartphone has multiple apps installed, we should not assume that because one app is authorized to discover certain services that means that all other apps on that smartphone are also authorized to discover those services. For example, just because a medical app on a smartphone is authorized to discover and communicate with the user's medical devices such as an implanted insulin monitor, that doesn't mean that social network apps or games on that same smartphone are also authorized to discover and communicate with those medical devices.

Note that when the text above talks about a user or app being "authorized" we're not talking about authorization controls being enforced by the laptop or smartphone. Controls enforced by the laptop or smartphone operating system are appropriate and have their place, but the kind of authorization controls we're talking about here are enforced by the entity being discovered. When the entity being discovered receives a query from an authorized source, it answers the query. When the entity being discovered receives a query from an unauthorized source, it does not answer the query. The important question is the granularity of the "source" referred to -- is it a physical device, a user, or an app? (This analysis presupposes that the host operating system on the device has sufficient memory protection and access controls to protect one user's secret key material from being accessed and abused by another user, or one app's secret key material from being accessed and abused by another app. For a device without such protection, only the per-device granularity of trust is applicable.)

4. Desirable Security Properties

For each of the operations and information levels described above, we need to consider what threats we are concerned about.

Authenticity & Integrity
: Can we trust the information we receive? Has it been modified in flight by an adversary? Do we trust the source of the information?

Confidentiality
: Who can read the information sent in messages? Ideally this should only be the appropriate trusted parties, but it can be hard to define who "the appropriate trusted parties" are. The "Discover" operation in particular is often used to discover new entities that the device did not previously know about. It may be tricky to work out how a device can have an established trust relationship with a new entity it has never previously communicated with.

Anonymity
: Does the information exchange reveal the identity of either participant? In this context "identity" can mean things like the name, email address, or phone number of the human user. It could mean things like the hostname or MAC address of the device. Even when information is authenticated and confidential, there can be unexpected sources of information leakage. For example, if suitable precautions are not taken, the source MAC address in data packets can reveal the identity of the device manufacturer, which can yield clues about the nature of the device.

Resistance to Dictionary Attacks
: It can be tempting to use simple one-way hash functions to obscure sensitive identifiers. This transforms a sensitive unique identifier such as an email address into a scrambled (but still unique) identifier. Unfortunately simple solutions may be vulnerable to offline dictionary attacks. Given a scrambled unique identifier, it may be possible to do a brute-force attack, trying billions of known and speculative email addresses until a match is found.

Resistance to Tracking
: In today’s world, we have to be sensitive to any unchanging unique identifier, no matter how thoroughly and irreversibly scrambled it may be. Even though an attacker may not be able to divine the origin of a scrambled unique identifier, the unchanging unique identifier may still be correlated with other things. If a given unchanging unique identifier appears on a cafe network every

morning when a certain person comes in to get coffee, then with some certainty that unchanging unique identifier can be associated with that person, and used to track their movements around the city for the rest of their workday. Consequently, in cases where this threat is a concern, all cleartext identifiers used on the network need to be rotated according to some policy, so that a given identifier is not reused for too long or in different locations. These changing identifiers can be decoded by trusted entities, but are meaningless to anyone else.

Resistance to Message Linking

Is it possible to link or correlate exchanges across discovery operations? For example, do Discovery messages reveal information about future Use messages, or vice versa? This can be done via sender MAC address, for example. An adversary can use linkability information to de-anonymize service users or providers, even in the event that, individually, no information leaks from any particular message alone (e.g., because it’s encrypted in transit). For example, even if persistent identifiers are rotated periodically, if all identifiers are not rotated in unison then the overlap period can be used to track the user across identifier rotations.

Resistance to Denial-of-Service Attack

In any protocol where the receiver of messages has to perform cryptographic operations on those messages, there is a risk of a brute-force flooding attack causing the receiver to expend excessive amounts of CPU time (and battery power) just processing and discarding those messages.

5. Other Operational Requirements

5.1. Power Management

Many modern devices, especially battery-powered devices, use power management techniques to conserve energy. One such technique is for a device to transfer information about itself to a proxy, which will act on behalf of the device for some functions, while the device itself goes to sleep to reduce power consumption. When the proxy determines that some action is required which only the device itself can perform, the proxy may have some way (such as Ethernet "Magic Packet") to wake the device.

In many cases, the device may not trust the network proxy sufficiently to share all its confidential key material with the proxy. This poses challenges for combining private discovery that relies on per-query cryptographic operations, with energy-saving techniques that rely on having (somewhat untrusted) network proxies answer queries on behalf of sleeping devices.

5.2. Protocol Efficiency

Creating a discovery protocol that has the desired security properties may result in a design that is not efficient. To perform the necessary operations the protocol may need to send and receive a large number of network packets. This may consume an unreasonable amount of network capacity (particularly problematic when it’s shared wireless spectrum), cause an unnecessary level of power consumption (particularly problematic on battery devices) and may result in the discovery process being slow.

It is a difficult challenge to design a discovery protocol that has the property of obscuring the details of what it is doing from unauthorized observers, while also managing to do that quickly and efficiently.

5.3. Secure Initialization

One of the challenges implicit in the preceding discussions is that whenever we discuss "trusted entities" versus "untrusted entities", there needs to be some way that trust is initially established, to convert an "untrusted entity" into a "trusted entity".

One way to establish trust between two entities is to trust a third party to make that determination for us. For example, the X.509 certificates used by TLS and HTTPS web browsing are based on the model of trusting a third party to tell us who to trust. There are some difficulties in using this model for establishing trust for

service discovery uses. If we want to print our tax returns or medical documents on "our" printer, then we need to know which printer on the network we can trust be be "our" printer. All of the printers we discover on the network may be legitimate printers made by legitimate printer manufacturers, but not all of them are "our" printer. A third-party certificate authority cannot tell us which one of the printers is ours.

Another common way to establish a trust relationship is Trust On First Use (TOFU), as used by ssh. The first usage is a Leap Of Faith, but after that public keys are exchanged and at least we can confirm that subsequent communications are with the same entity. In today's world, where there may be attackers present even at that first use, it would be preferable to be able to establish a trust relationship without requiring an initial Leap Of Faith.

Techniques now exist for securely establishing a trust relationship without requiring an initial Leap Of Faith. Trust can be established securely using a short passphrase or PIN with cryptographic algorithms such as Secure Remote Password (SRP) [RFC5054] or a Password Authenticated Key Exchange like J-PAKE [RFC8236] using a Schnorr Non-interactive Zero-Knowledge Proof [RFC8235].

Such techniques require a user to enter the correct passphrase or PIN in order for the cryptographic algorithms to establish working communication. This avoids the human tendency to simply press the "OK" button when asked if they want to do something on their electronic device. It removes the human fallibility element from the equation, and avoids the human users inadvertently sabotaging their own security.

Using these techniques, if a user tries to print their tax return on a printer they've never used before (even though the name looks right) they'll be prompted to enter a pairing PIN, and the user *cannot* ignore that warning. They can't just press an "OK" button. They have to walk to the printer and read the displayed PIN and enter it. And if the intended printer is not displaying a pairing PIN, or is displaying a different pairing PIN, that means the user may be being spoofed, and the connection will not succeed, and the failure will not reveal any secret information to the attacker. As much as the human desires to "just give me an OK button to make it print" (and the attacker desires them to click that OK button too) the cryptographic algorithms do not give the user the ability to opt out of the security, and consequently do not give the attacker any way to persuade the user to opt out of the security protections.

6. Informative References

[RFC5054] Taylor, D., Wu, T., Mavrogiannopoulos, N., and T. Perrin, "Using the Secure Remote Password (SRP) Protocol for TLS Authentication", RFC 5054, DOI 10.17487/RFC5054, November 2007, <https://www.rfc-editor.org/info/rfc5054>.

[RFC8235] Hao, F., Ed., "Schnorr Non-interactive Zero-Knowledge Proof", RFC 8235, DOI 10.17487/RFC8235, September 2017, <https://www.rfc-editor.org/info/rfc8235>.

[RFC8236] Hao, F., Ed., "J-PAKE: Password-Authenticated Key Exchange by Juggling", RFC 8236, DOI 10.17487/RFC8236, September 2017, <https://www.rfc-editor.org/info/rfc8236>.

Author's Address

Stuart Cheshire
Apple Inc.
1 Infinite Loop
Cupertino, California 95014
USA

Phone: +1 408 974 3207
Email: cheshire@apple.com