Internet Engineering Task Force (IETF)
Request for Comments: 8117
Category: Informational
ISSN: 2070-1721

C. Huitema
Private Octopus Inc.
D. Thaler
Microsoft
R. Winter
University of Applied Sciences Augsburg

March 2017

Current Hostname Practice Considered Harmful

Abstract

Giving a hostname to your computer and publishing it as you roam from one network to another is the Internet's equivalent of walking around with a name tag affixed to your lapel. This current practice can significantly compromise your privacy, and something should change in order to mitigate these privacy threats.

There are several possible remedies, such as fixing a variety of protocols or avoiding disclosing a hostname at all. This document describes some of the protocols that reveal hostnames today and sketches another possible remedy, which is to replace static hostnames by frequently changing randomized values.

Status of This Memo

Table of Contents

1. Introduction

There is a long established practice of giving names to computers. In the Internet protocols, these names are referred to as "hostnames" [RFC7719]. Hostnames are normally used in conjunction with a domain name suffix to build the Fully Qualified Domain Name (FQDN) of a host [RFC1983]. However, it is common practice to use the hostname without further qualification in a variety of applications from file sharing to network management. Hostnames are typically published as part of domain names and can be obtained through a variety of name lookup and discovery protocols.

Hostnames have to be unique within the domain in which they are created and used. They do not have to be globally unique identifiers, but they will always be at least partial identifiers, as discussed in Section 3.

The disclosure of information through hostnames creates a problem for mobile devices. Adversaries that monitor a remote network such as a Wi-Fi hot spot can obtain the hostname through passive monitoring or active probing of a variety of Internet protocols, such as DHCP or Multicast DNS (mDNS). They can correlate the hostname with various other information extracted from traffic analysis and other information sources, and they can potentially identify the device, device properties, and its user [TRAC2016].

2. Naming Practices

There are many reasons to give names to computers. This is particularly true when computers operate on a network. Operating systems like Microsoft Windows or Unix assume that computers have a "hostname." This enables users and administrators to do things such as ping a computer, add its name to an access control list, remotely mount a computer disk, or connect to the computer through tools such as telnet or remote desktop. Other operating systems maintain multiple hostnames for different purposes, e.g., for use with certain protocols such as mDNS.

In most consumer networks, naming is pretty much left to the discretion of the user. Some will pick names of planets or stars, others will pick names of fruits or flowers, and still others will pick whatever suits their mood when they unwrap the device. As long as users are careful to not pick a name already in use on the same network, anything goes. Very often, however, the operating system suggests a hostname at the time of installation, which can contain the user name, the login name, and information learned from the device itself such as the brand, model, or maker of the device [TRAC2016].

In large organizations, collisions are more likely and a more structured approach is necessary. In theory, organizations could use multiple DNS subdomains to ease the pressure on uniqueness, but in practice many don’t and insist on unique flat names, if only to simplify network management. To ensure unique names, organizations will set naming guidelines and enforce some kind of structured naming. For example, within the Microsoft corporate network, computer names are derived from the login name of the main user, which leads to names like "huitema-test2" for a machine that one of the authors used to test software.

There is less pressure to assign names to small devices including, for example, smart phones, as these devices typically do not enable sharing of their disks or remote login. As a consequence, these devices often have manufacturer-assigned names, which vary from generic names like "Windows Phone" to completely unique names like "BrandX-123456-7890-abcdef" and often contain the name of the device owner, the device’s brand name, and often also a hint as to which language the device owner speaks [TRAC2016].

3. Partial Identifiers

Suppose an adversary wants to track the people connecting to a specific Wi-Fi hot spot, for example, in a railroad station. Assume that the adversary is able to retrieve the hostname used by a specific laptop. That, in itself, might not be enough to identify the laptop’s owner. Suppose, however, that the adversary observes that the laptop name is "dthaler-laptop" and that the laptop has established a VPN connection to the Microsoft corporate network. The two pieces of information, put together, firmly point to Dave Thaler, employed by Microsoft. The identification is successful.

In the example, we saw a login name inside the hostname, and that certainly helped identification. But generic names like "jupiter" or "rosebud" also provide partial identification, especially if the adversary is capable of maintaining a database recording, among other information, the hostnames of devices used by specific users. Generic names are picked from vocabularies that include thousands of potential choices. Finding the name reduces the scope of the search significantly. Other information such as the visited sites will quickly complement that data and can lead to user identification.

Also, the special circumstances of the network can play a role. Experiments on operational networks such as the IETF meeting network have shown that, with the help of external data such as the publicly available IETF attendees list or other data sources such as

Lightweight Directory Access Protocol (LDAP) servers on the network [TRAC2016], the identification of the device owner can become trivial given only partial identifiers in a hostname.

Unique names assigned by manufacturers do not directly encode a user identifier, but they have the property of being stable and unique to the device in a large context. A unique name like "BrandX-123456-7890-abcdef" allows efficient tracking across multiple domains. In theory, this only allows tracking of the device but not of the user. However, an adversary could correlate the device to the user through other means, for example, the one-time capture of some cleartext traffic. Adversaries could then maintain databases linking a unique hostname to a user identity. This will allow efficient tracking of both the user and the device.

4. Protocols That Leak Hostnames

Many IETF protocols can leak the "hostname" of a computer. A non-exhaustive list includes DHCP, DNS address to name resolution, Multicast DNS, Link-local Multicast Name Resolution, and DNS service discovery.

4.1. DHCP

Shortly after connecting to a new network, a host can use DHCP [RFC2131] to acquire an IPv4 address and other parameters [RFC2132]. A DHCP query can disclose the "hostname." DHCP traffic is sent to the broadcast address and can be easily monitored, enabling adversaries to discover the hostname associated with a computer visiting a particular network. DHCPv6 [RFC3315] shares similar issues.

The problems with the hostname and FQDN parameters in DHCP are analyzed in [RFC7819] and [RFC7824]. Possible mitigations are described in [RFC7844].

4.2. DNS Address to Name Resolution

The domain name service design [RFC1035] includes the specification of the special domain "in-addr.arpa" for resolving the name of the computer using a particular IPv4 address, using the PTR format defined in [RFC1033]. A similar domain, "ip6.arpa", is defined in [RFC3596] for finding the name of a computer using a specific IPv6 address.

Adversaries who observe a particular address in use on a specific network can try to retrieve the PTR record associated with that address and thus the hostname of the computer, or even the FQDN of that computer. The retrieval may not be useful in many IPv4 networks due to the prevalence of NAT, but it could work in IPv6 networks. Other name lookup mechanisms, such as [RFC4620], share similar issues.

4.3. Multicast DNS

Multicast DNS (mDNS) is defined in [RFC6762]. It enables hosts to send DNS queries over multicast and to elicit responses from hosts participating in the service.

If an adversary suspects that a particular host is present on a network, the adversary can send mDNS requests to find, for example, the A or AAAA records associated with the hostname in the ".local" domain. A positive reply will confirm the presence of the host.

When a new responder starts, it must send a set of multicast queries to verify that the name that it advertises is unique on the network and to populate the caches of other mDNS hosts. Adversaries can monitor this traffic and discover the hostname of computers as they join the monitored network.

mDNS further allows queries to be sent via unicast to port 5353. An adversary might decide to use unicast instead of multicast in order to hide from, e.g., intrusion detection systems.

4.4. Link-Local Multicast Name Resolution

Link-Local Multicast Name Resolution (LLMNR) is defined in [RFC4795]. The specification did not achieve consensus as an IETF standard, but it is widely deployed. Like mDNS, it enables hosts to send DNS queries over multicast and to elicit responses from computers implementing the LLMNR service.

Like mDNS, LLMNR can be used by adversaries to confirm the presence of a specific host on a network by issuing a multicast request to find the A or AAAA records associated with the hostname in the ".local" domain.

When an LLMNR responder starts, it sends a set of multicast queries to verify that the name that it advertises is unique on the network. Adversaries can monitor this traffic and discover the hostname of computers as they join the monitored network.

4.5. DNS-Based Service Discovery

DNS-based Service Discovery (DNS-SD) is described in [RFC6763]. It enables participating hosts to retrieve the location of services proposed by other hosts. It can be used with DNS servers or in conjunction with mDNS in a serverless environment.

Participating hosts publish a service described by an "instance name", which is typically chosen by the user responsible for the publication. While this is obviously an active disclosure of information, privacy aspects can be mitigated by user control. Services should only be published when deciding to do so, and the information disclosed in the service name should be well under the control of the device's owner.

In theory, there should not be any privacy issue, but in practice the publication of a service also forces the publication of the hostname due to a chain of dependencies. The service name is used to publish a PTR record announcing the service. The PTR record typically points to the service name in the local domain. The service names, in turn, are used to publish TXT records describing service parameters and SRV records describing the service location.

SRV records are described in [RFC2782]. Each record contains four parameters: priority, weight, port number, and hostname. While the service name published in the PTR record is chosen by the user, the "hostname" in the SRV record is indeed the hostname of the device.

Adversaries can monitor the mDNS traffic associated with DNS-SD and retrieve the hostname of computers advertising any service with DNS-SD.

4.6. NetBIOS-over-TCP

Amongst other things, NetBIOS-over-TCP [RFC1002] implements a name registration and resolution mechanism called the NetBIOS Name Service. In practice, NetBIOS resource names are often based on hostnames.

NetBIOS allows an application to register resource names and to resolve such names to IP addresses. In environments without a NetBIOS Name Server, the protocol makes extensive use of broadcasts from which resource names can be easily extracted. NetBIOS also allows querying for the names registered by a node directly (node status).

5. Randomized Hostnames as a Remedy

There are several ways to remedy the hostname practices. We could instruct people to just turn off any protocol that leaks hostnames, at least when they visit some "insecure" place. We could also examine each particular standard that publishes hostnames and somehow fix the corresponding protocols. Or, we could attempt to revise the way devices manage the hostname parameter.

There is a lot of merit in turning off unneeded protocols when visiting insecure places. This amounts to attack-surface reduction and is clearly beneficial -- this is an advantage of the stealth mode defined in [RFC7288]. However, there are two issues with this advice. First, it relies on recognizing which networks are secure or insecure. This is hard to automate, but relying on end-user judgment may not always provide good results. Second, some protocols such as DHCP cannot be turned off without losing connectivity, which limits the value of this option. Also, the services that rely on protocols that leak hostnames such as mDNS will not be available when switched off. In addition, not always are hostname-leaking protocols well-known, as they might be proprietary and come with an installed application instead of being provided by the operating system.

It may be possible in many cases to examine a protocol and prevent it from leaking hostnames. This is, for example, what is attempted for DHCP in [RFC7844]. However, it is unclear that we can identify, revisit, and fix all the protocols that publish hostnames. In particular, this is impossible for proprietary protocols.

We may be able to mitigate most of the effects of hostname leakage by revisiting the way platforms handle hostnames. In a way, this is similar to the approach of Media Access Control (MAC) address randomization described in [RFC7844]. Let’s assume that the operating system, at the time of connecting to a new network, picks a random hostname and starts publicizing that random name in protocols such as DHCP or mDNS, instead of the static value. This will render monitoring and identification of users by adversaries much more difficult without preventing protocols such as DNS-SD from operating as expected. This, of course, has implications on the applications making use of such protocols, e.g., when the hostname is being displayed to users of the application. They will not as easily be able to identify, e.g., network shares or services based on the hostname carried in the underlying protocols. Also, the generation of new hostnames should be synchronized with the change of other tokens used in network protocols such as the MAC or IP address to prevent correlation of this information. For example, if the IP address changes but the hostname stays the same, the new IP address can be correlated to belong to the same device based on a leaked hostname.

Some operating systems, including Windows, support "per network" hostnames, but some other operating systems only support "global" hostnames. In that case, changing the hostname may be difficult if the host is multihomed, as the same name will be used on several networks. Other operating systems already use potentially different hostnames for different purposes, which might be a good model to combine both static hostnames and randomized hostnames based on their potential use and threat to a user’s privacy.

Obviously, further studies are required before the idea of randomized hostnames can be implemented.

6. Security Considerations

This document does not introduce any new protocol. It does point to potential privacy issues in a set of existing protocols.

There are obvious privacy gains to changing to randomized hostnames and also to changing these names frequently. However, wide deployment might affect security functions or current practices. For example, incident response using hostnames to track the source of traffic might be affected. It is common practice to include hostnames and reverse lookup information at various times during an investigation.

7. IANA Considerations

This document does not require any IANA actions.

8. Informative References

[RFC1002] NetBIOS Working Group in the Defense Advanced Research Projects Agency, Internet Activities Board, and End-to-End Services Task Force, "Protocol standard for a NetBIOS service on a TCP/UDP transport: Detailed specifications", STD 19, RFC 1002, DOI 10.17487/RFC1002, March 1987, <http://www.rfc-editor.org/info/rfc1002>.

[RFC1033] Lottor, M., "Domain Administrators Operations Guide", RFC 1033, DOI 10.17487/RFC1033, November 1987, <http://www.rfc-editor.org/info/rfc1033>.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <http://www.rfc-editor.org/info/rfc1035>.

[RFC1983] Malkin, G., Ed., "Internet Users' Glossary", FYI 18, RFC 1983, DOI 10.17487/RFC1983, August 1996, <http://www.rfc-editor.org/info/rfc1983>.

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, DOI 10.17487/RFC2131, March 1997, <http://www.rfc-editor.org/info/rfc2131>.

[RFC2132] Alexander, S. and R. Droms, "DHCP Options and BOOTP Vendor Extensions", RFC 2132, DOI 10.17487/RFC2132, March 1997, <http://www.rfc-editor.org/info/rfc2132>.

[RFC2782] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, DOI 10.17487/RFC2782, February 2000, <http://www.rfc-editor.org/info/rfc2782>.

[RFC3315] Droms, R., Ed., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, DOI 10.17487/RFC3315, July 2003, <http://www.rfc-editor.org/info/rfc3315>.

[RFC3596] Thomson, S., Huitema, C., Ksinant, V., and M. Souissi, "DNS Extensions to Support IP Version 6", RFC 3596, DOI 10.17487/RFC3596, October 2003, <http://www.rfc-editor.org/info/rfc3596>.

[RFC4620] Crawford, M. and B. Haberman, Ed., "IPv6 Node Information Queries", RFC 4620, DOI 10.17487/RFC4620, August 2006, <http://www.rfc-editor.org/info/rfc4620>.

[RFC4795] Aboba, B., Thaler, D., and L. Esibov, "Link-local Multicast Name Resolution (LLMNR)", RFC 4795, DOI 10.17487/RFC4795, January 2007, <http://www.rfc-editor.org/info/rfc4795>.

[RFC6762] Cheshire, S. and M. Krochmal, "Multicast DNS", RFC 6762, DOI 10.17487/RFC6762, February 2013, <http://www.rfc-editor.org/info/rfc6762>.

[RFC6763] Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", RFC 6763, DOI 10.17487/RFC6763, February 2013, <http://www.rfc-editor.org/info/rfc6763>.

[RFC7288] Thaler, D., "Reflections on Host Firewalls", RFC 7288, DOI 10.17487/RFC7288, June 2014, <http://www.rfc-editor.org/info/rfc7288>.

[RFC7719] Hoffman, P., Sullivan, A., and K. Fujiwara, "DNS Terminology", RFC 7719, DOI 10.17487/RFC7719, December 2015, <http://www.rfc-editor.org/info/rfc7719>.

[RFC7819] Jiang, S., Krishnan, S., and T. Mrugalski, "Privacy Considerations for DHCP", RFC 7819, DOI 10.17487/RFC7819, April 2016, <http://www.rfc-editor.org/info/rfc7819>.

[RFC7824] Krishnan, S., Mrugalski, T., and S. Jiang, "Privacy Considerations for DHCPv6", RFC 7824, DOI 10.17487/RFC7824, May 2016, <http://www.rfc-editor.org/info/rfc7824>.

[RFC7844] Huitema, C., Mrugalski, T., and S. Krishnan, "Anonymity Profiles for DHCP Clients", RFC 7844, DOI 10.17487/RFC7844, May 2016, <http://www.rfc-editor.org/info/rfc7844>.

[TRAC2016] Faath, M., Winter, R., and F. Weisshaar, "How Broadcast Data Reveals Your Identity and Social Graph", IEEE, Wireless Communications and Mobile Computing Conference (IWCMC), 2016 International, DOI 10.1109/IWCMC.2016.7577084, September 2016.

Acknowledgments

Thanks to the members of the INTAREA Working Group for discussions and reviews.

Authors' Addresses

Christian Huitema
Private Octopus Inc.
Friday Harbor, WA 98250
United States of America

Email: huitema@huitema.net

Dave Thaler
Microsoft
Redmond, WA 98052
United States of America

Email: dthaler@microsoft.com

Rolf Winter
University of Applied Sciences Augsburg
Augsburg
DE

Email: rolf.winter@hs-augsburg.de