Autoconfiguration for IP Networking:

Enabling Local Communication

Erik Guttman • *Sun Microsystems, Germany*

> It would be ideal if a host implementation of the Internet protocol suite could be entirely self-configuring. This would allow the whole suite to be implemented in ROM or cast into silicon, it would simplify diskless workstations, and it would be an immense boon to harried LAN administrators as well as system vendors. We have not reached this ideal; in fact, we are not even close. —RFC 1122[1]

IP hosts and network infrastructure have historically been difficult to configure — requiring network services and relying on highly trained network administrators — but emerging networking protocols promise to enable hosts to establish IP networks without prior configuration or network services. Even very simple devices with few computing resources will be able to communicate via standards-track protocols wherever they are attached. Current IETF standardization efforts, such as those in the Zeroconf working group, aim to make this form of networking simple and inexpensive.

Hosts that are permanently connected to an administered network are usually assigned *static* network configurations by network administrators. Other hosts are attached to administered networks (such as corporate local-area networks or dial-in accounts) using *dynamic* network configuration. In these, all necessary parameters are assigned to the host by a network configuration service, which also requires configuration. In many situations — impromptu meetings, administered network misconfigurations, or network service failures, for example — establishing an IP network is desirable, but administering it can be impractical or impossible. In these cases, *automatic* network configuration of parameters is valuable for hosts. The IETF Zeroconf WG's real goal is to enable direct communications between two or more computing devices via IP. In this tutorial, I examine the background, current status, and future prospects for zero configuration networking.

Zero Configuration Networking

Automatic configuration parameters have different properties from those assigned by static and dynamic configuration. They are ephemeral; they will likely be different each time they are obtained and might even change at any time. Automatically configured hosts actively participate in assigning and maintaining their configuration parameters, which have only local significance. Autonomy from network services implies that hosts must network configuration.

In direct contrast, normal IP configuration is persistent (especially for servers), or at the very least, stable. The IP protocol suite aims at scalability, especially with respect to configuration. Addresses and names often have global significance, which has proven essential for enabling Internet growth. Obtaining and managing global addresses and names requires a great deal of administrative work, however. These processes are not at all automatic and likely never will be.

Despite these differences, the essential zero configuration networking protocols really imply changes to only the lower layers of IP-enabled devices. (See the sidebar "IP Host Layering", next page, for an introduction to the terminology required for discussing automatic configuration.)

Existing network-based applications will work without modification over enhanced network service and application layers using standard interfaces. Indeed, users should not even be aware that the network service layer has been configured automatically rather than statically or dynamically.

Four functions will benefit from zero configu-

IP Host Layering

Layering provides the foundation for numerous, stable extensible computing platforms. Figure A depicts the pervasive layered architecture, often called the *IP stack*, which is used for Internet hosts. This roughly corresponds to the OSI seven-layer model.[1] The figure excludes the OSI presentation and session layers. IP applications implement data presentation functions themselves. Session features such as encryption, compression, or persistence between protocol transactions are added in an ad hoc fashion at various layers.

Each layer provides services to the layer above it through standard interfaces. If lower layers provide the same functionality using the same interfaces, services can be implemented in different ways; new mechanisms defined at the network services layer can thus support unmodified existing applications. Avoiding changes in upper layers eases the adoption of new Internet technologies. Network service layer enhancements that require client applications (such as e-mail readers and Web browsers) to be upgraded are not broadly adopted.

Each layer could be automatically configured. In practice, the less configuration required, the better, because simpler technology works more predictably and eases deployment. The transport service, link control, and media access layers rarely require configuration in Internet hosts. By contrast, the application and network service layers nearly always require configuration in order to operate at all.

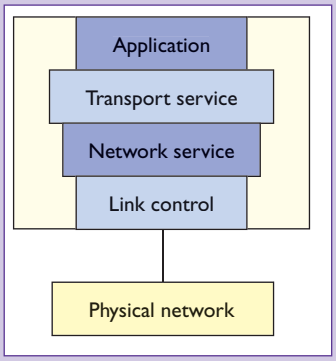

Figure A. Internet host layering. Zero configuration protocols will be implemented at the application and network service layers of the Internet protocol stack.

References

1. A. Tanenbaum, *Computer Networks*, second edition, Prentice-Hall, Englewood Cliffs, N. J., 1989.

ration protocols, in the context of both IPv4 and IPv6. With no modification to existing interfaces, zero configuration protocols will improve *name-to-address translation* (at the application level) and *IP interface configuration* (at the network level). Functions previously unavailable to IP hosts will introduce new interfaces: *service discovery* at the application layer and *multicast address allocation* at the network layer.

These additional services will not disrupt existing applications. They will "raise the bar" by providing additional features long absent from the Internet protocol suite, but (in the case of service discovery) available in proprietary network protocol suites from Apple, Microsoft, and Novell. (See the sidebar "Early Autoconfiguration Efforts"). These proprietary protocols continue to be used only because of their ease-of-configuration features. Adopting emerging zero configuration protocol standards will let us retire proprietary networking — a move that has broad support. Even network equipment vendors uniformly accept that proprietary network protocols have seen their day and should be replaced by IP standards.

For reasons of scalability and reducing impact on existing networks, the zero configuration protocols' effect on the overall network must be limited. The algorithms used for zero configuration protocols generally use multicast. In practice, these protocols are limited to either a single network link (that is, routers do not forward these protocol messages) or to a set of networks (where some routers are configured as boundaries, over which protocol messages are not forwarded).

Defining an Approach

Those working on IETF zero configuration protocol standardization (currently in the Zeroconf, Service Location Protocol, DNS Extensions, and IPng working groups) have considered two main approaches to overcoming the differences between configured and automatic operation.

The first strategy requires transitions between local and global configuration and has been explored through consumer-oriented operating system software since 1998. This strategy implies that hosts would support automatic configuration only for as long as they lacked global configuration. The two modes are exclusive, and the presence of a dynamic configuration service requires a transition from automatic (local) to dynamic (global) configuration.

An example of this transition strategy is the network interface autoconfiguration protocol adopted for desktop operating systems from Apple and Microsoft. This protocol (which the IETF has not yet standardized) enables a host to simply choose an unassigned IP address from a reserved range. The host then attempts to obtain (global) IP configuration parameters from the network via the Dynamic host configuration protocol.[2] The host issues periodic DHCP requests, which will eventually succeed in reaching a DHCP server if one ever becomes available on the network. Once a DHCP server responds and offers IP configuration parameters, these replace automatic configuration.

This mechanism works fine for clients employing common client-server protocols because very few make use of long-duration connections. Individual network application operations result in distinct transactions even when connections fail. If the client host experi-

ences network reconfiguration, applications simply establish new connections.

If a server's configuration changes, however, recovery is not so easy: Client applications cease to function if they cannot find a server. Servers with dynamic addresses can only be located via a dynamic service discovery protocol, and very few IP-based applications currently employ service discovery.

Server address reconfiguration can break server software, which typically binds to a (presumably immutable) address to accept incoming messages. Moreover, when a server reconfigures via DHCP, it can no longer communicate with clients that have not yet reconfigured. Conversely, if DHCP configures client systems and then fails to configure a server (if the DHCP server becomes unreachable, for example), the clients with global parameters will be unable to communicate with the server, which still has only local, automatic configuration.

Finally, some extremely simple devices might support only local IP configuration and would be unable to communicate with hosts reconfigured using DHCP. Very cheap appliances could be developed to support remote monitoring and control services, for example, and clients would need local IP configuration in order to communicate with these devices at all, unless additional network infrastructure is available.

Given the problems that arise from IP configuration transition, the Zeroconf WG now discourages the transitioning approach. Hosts should either use automatic configuration alone, or *in addition to* dynamic or static configuration. Two hosts attached to the same network implementing zero configuration protocols will be able to communicate regardless of whether DHCP or any other servers are available. They will only need to reconfigure their addresses (or possibly their names) in the event of a conflict.

Emerging Solutions

The Zeroconf WG has defined requirements for four zero configuration networking protocol areas. For further explanation and details, please refer to the draft requirements document.[3] Current IETF efforts have produced standards, or soon will, in each of the following.

Address Autoconfiguration

The first protocol area is *address autoconfiguration*. For an IP stack to deliver IP messages, each communicating endpoint (source and destination) requires a unique IP address within the scope in which the address will be used. A link-local address, for example, is configured to be unique on the link. Address autoconfiguration requirements include allowing a host to

- configure its interfaces with unique addresses;
- determine which subnet mask to use (the subnet mask identifies the network address and, among other things, allows an IP stack to determine whether it can deliver a datagram directly);
- detect duplicate address assignment; and
- cope with collisions.

The current IPv4 link-local autoconfiguration specification[4] is backward compatible with Apple Macintosh and Microsoft Windows operating system software with one exception: It recommends maintaining autoconfigured addresses (even if an interface is also configured with a global IPv4 address) rather than transitioning from local to global. The current specification also includes guidelines to help disambiguate link-local addressing for hosts with more than one IP-enabled interface. Link-local autoconfiguration for IPv6 is an IETF draft standard.[5]

Early Autoconfiguration Efforts

The Internet protocol suite emerged as the data communication standard for a network run by and for researchers. Their design goals were primarily interoperability, extensibility, and scalability. IP networks achieved growth and enabled global communication by using unique parameters (for naming, addressing, and so on) and mechanisms for delegating their administration.

At the same time, network software (and equipment) vendors were developing proprietary protocol suites that stressed ease of use and deployment. The success of Apple's AppleTalk,[1] Novell's IPX,[2] and Microsoft's NetBIOS/SMB[3] arose from their automatic address configuration capabilities, decentralized service discovery, and naming functions, which facilitated local communication and sharing of resources such as files and printers.

Zero configuration will bridge the gap between these two distinct families of protocols. IP-enabled hosts and applications will be able to take advantage of mechanisms similar to those provided by AppleTalk. It is not possible, however, to completely automate the configuration of the Internet. Zero configuration protocols allow local communication on networks of a limited scale (defined functionally rather than absolutely). When automatic configuration no longer suffices, administrators must plan and deploy scalable configured networks.

References

1. S. Gursharan et al., *Inside AppleTalk*, Addison-Wesley, Reading, Mass., 1990.
2. *IPX RIP and SAP Router Specification, version 1.30*, Part Number 107-000029-001, Novell Inc., May 1996.
3. *SMB File Sharing Protocol Extensions 3.0, version 1.09*, Microsoft Networks, Nov. 1989.

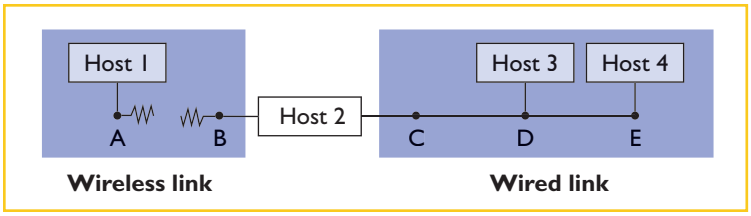

Figure 1. IP address autoconfiguration. Host addresses (A through E) are unique for each shared wired or wireless link.

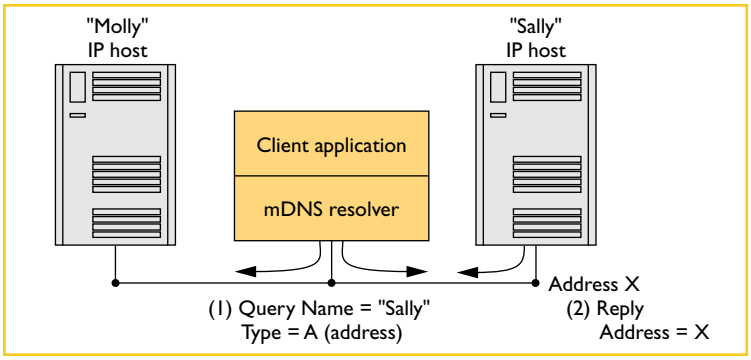

Figure 2. Multicast DNS protocol interaction. The client obtains the IP address for the host named "Sally" by issuing a request to a well-known multicast address and awaiting a reply.

Each interface of each device in Figure 1 can obtain and maintain a unique address assignment. Host 1 and host 2 share a wireless link, upon which addresses A and B are distinct. Hosts 2, 3, and 4 share a wired link upon which addresses C, D, and E are unique.

To reduce confusion, Host 2 will not allocate addresses that conflict with an assigned link-local address on any link to which it is attached. Host 2 will not attempt to allocate address A on the wired link because A has already been allocated on the wireless link. Because host 2 has no control over others, such as hosts 1 and 3, address A could be the same as address D. Such a situation could lead to ambiguities for host 2. This complicates support for link-local address autoconfiguration of hosts with multiple interfaces.

Name-to-Address Translation

The second zero configuration protocol area is *name-to-address translation*. IP applications typically identify endpoints on the network by name rather than by address. This provides operational stability when an endpoint's address changes because its name remains the same. A zero configuration protocol for name-to-address translation requires mechanisms for:

- obtaining the IP address associated with a name, and
- determining the name associated with an IP address.

This latter feature facilitates communication with the client in the future and enables the server to generate human-readable log entries.

Link-local multicast DNS[6] is defined for use over IPv4 and IPv6 to locally resolve names using the DNS protocol, but without requiring a dedicated DNS server. The node information query protocol can also be used for name-to-address translation over IPv6.[7]

Figure 2 illustrates name-to-address resolution using multicast DNS. A client application requests the address corresponding to the name "Sally," which can be translated to an address by issuing a DNS request to a well-known multicast IP address. Each host listens for these requests and responds if the interface on which the multicast DNS request was received is configured with the name requested.

Service Discovery

The third zero configuration protocol area is *service discovery*. Clients should be able to discover services on the network without prior configuration, and without any administered configuration management services (such as directories) on the network. Furthermore, the service discovery protocol must not cause broadcast storms or other unscalable behavior. (Some existing service discovery protocols — most notably the Service Advertising Protocol from the IPX protocol suite — require inordinate network resources.)

Some services, such as generic Web proxies, DNS servers, or SMTP relays, are indistinct; that is, any server of that type will perform the exact same function. Other services, such as nonreplicated databases, file servers, or IP-enabled printers, are distinct in that each instance of the service is unique. The ability to distinguish between servers of the latter type lets a client discover the server it needs, rather than all the servers it can communicate with (most of which will be useless to it).

The IETF has standardized two mechanisms for service discovery over IP networks. Version 2 of the service location protocol[8] (SLP) uses administrative-scope multicast because it was designed to scale up to a single administration (usually comprising an entire site, such as a campus or enterprise network). Most other zero configuration protocols are being defined for use with link-local multicast. SLP provides service discovery both by service type and attribute, so a client can find a distinct server by specifying required characteristics. SLP is defined for use over IPv4 and

IPv6.[9] (I discussed SLP in greater detail in an earlier IC article.[10])

The second IETF service-discovery mechanism is the DNS SRV Resource Record,[11] which allows clients to look up services via DNS. Clients specify the type of service, the transport protocol, and the domain name to look up. The reply to this query supplies a list of hosts that match the request.

Figure 3 illustrates service discovery using SLPv2. The client application requests the location of a service "Bar," including the type of service and service attributes required. The SLPv2 user agent multicasts this request on behalf of the client. SLPv2 service agents respond with a *service reply* if they are advertising services that match the request.

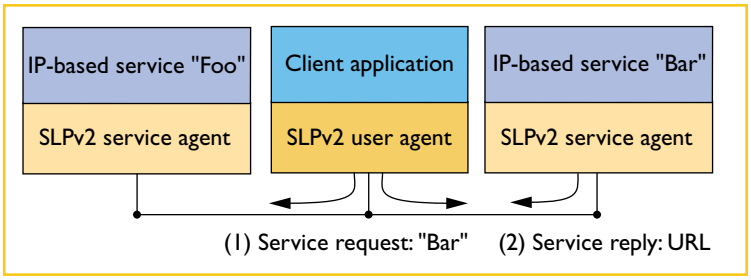

Figure 3. Service discovery using SLPv2. A client application requests the location of a service "Bar," and SLPv2 service agents respond with a service reply if they are advertising services that match the request.

Multicast Address Allocation

The fourth protocol area the working group identified is *multicast address allocation*. Some multicast-based applications need to obtain a unique multicast address to prevent other applications (or sessions based on the same application) from conflicting with them. A multicast address conflict can cause applications to fail in an analogous way to two hosts configured with the same IP address: Communication from the two distinct sessions could be delivered to incorrect destinations.

The zeroconf multicast address allocation protocol (ZMAAP)[12] allows applications to

- allocate unique addresses and maintain them over time;
- prevent reallocation of assigned addresses; and
- be notified of multicast allocation collision.

These requirements differ from those for address autoconfiguration because multicast addresses are a shared resource. In many cases, different processes present across the network need shared control of the allocation. The second and third ZMAAP requirements enable any process to prolong a session and to discover whether the session is still valid.

The current version of a proposed protocol specification that fulfills these requirements can be found at the Zeroconf working group page.

Securing Zeroconf Protocols

Proprietary autoconfiguration protocols provide no mechanisms for securing their basic operation. AppleTalk, IPX, and NetBIOS networking security mechanisms provide applications with user and group-oriented access control, but the automatic configuration protocols (address assignment, name resolution, and service discovery) remain insecure. This has kept these protocol suites simple, but has also left them vulnerable to attack.

Autoconfiguration can be dangerous. Without proper security mechanisms in place, anyone with access to a LAN can easily subvert the zero configuration protocols used there. As wireless networking technology deployment and shared-access networks expand across video cable, power line, and other residential-access media, unauthorized access becomes increasingly likely.

Running counter to the basic goals of zero configuration networking, security cannot be automatically configured, but it should still be simple to administer. The Zeroconf WG has identified a set of requirements to ensure that operation using zero configuration protocols will be no less secure than using their correlate configured IETF protocols.[3] Specific mechanisms for simply configuring IP hosts to operate securely (fulfilling these requirements) have been discussed, but no specification has yet been proposed.

An important area for further development is secure remote access to autoconfigured networks. Many home networking products already offer proprietary solutions for remotely controlling or monitoring devices in private networks. It might be useful to establish a standardized gateway mechanism that gives hosts on remote networks access to locally configured devices. Further study will also be required for creating easily administered and interoperable mechanisms for configuring security parameters in a group of devices.

Network Administration

Because zero configuration protocols allow hosts to be configured automatically and administratively at the same time, network administrators could face several new issues.

- IP hosts using locally assigned addresses and names might be accessible only on a single

LAN. This contrasts with hosts configured with addresses with a greater scope, such as globally unique IP addresses, which (ideally) are accessible from any network in the Internet (ignoring subtleties like firewalls).

- Administrators today expect network communications to be constrained to hosts they configure and control. In the future, much of the communication on individual links might be between devices with entirely local parameters — sometimes between devices that will never obtain administratively assigned parameters.
- On networks without DHCP, users expect administrators to enable networking through configuration assignments. This will change as users become accustomed to automatically configured networks.
- Users will want access to link-local configured services from anywhere on the network. This will require either additional configuration for the services or application-layer gateways, proxies, or a more complex strategy involving remote access to the network where link-local-only services reside.

Zero configuration protocols will likely simplify network administration by reducing problems with setting up and operating IP hosts for local communication. At the same time, hosts will potentially have twice as many configurations (local and global), which will give rise to complex situations. I have explored many of the architectural issues of zero configuration networking further elsewhere.[13]

Future Prospects

The IETF will publish the zero configuration protocol requirements specification and the emerging standards-track protocols soon. This will herald the development of simple interoperable IP-enabled devices and greatly increase LAN stability and usability.

There is also discussion of creating a profile to specify the set of zero configuration protocols that conforming hosts implement. This approach, inspired by the IP host requirements specification,[1] would motivate vendors to implement a single set of protocols. The IETF generally avoids recommending or requiring specific protocols (nearly all IETF standards are classified as *elective*). IP host requirements were published to describe prior experience rather than to prescribe a future solution. Thus, it is still unclear whether the Zeroconf working group will produce the profile document.

As I mentioned before, security mechanisms will require additional investigation, and new network administration challenges will likely arise, but IETF zero configuration protocols will soon be available; in fact, many already are.

References

1. R. Braden, "Requirements for Internet Hosts — Communication Layers," IETF RFC 1122, Oct. 1989; available at http://www.rfc-editor.org/rfc/rfc1122.txt.
2. R. Droms, "Dynamic Host Configuration Protocol," IETF RFC 2131, Mar. 1997; available at http://www.rfc-editor.org/rfc/rfc2131.txt.
3. M. Hattig, "ZeroConf Requirements," Internet draft, Zeroconf WG, Mar. 2001; work in progress.
4. S. Cheshire and B. Aboba, "Dynamic Configuration of IPv4 Link-local Addresses," Internet draft, Zeroconf WG, Mar. 2001; work in progress.
5. S. Thomson and T. Narten, "IPv6 Stateless Address Autoconfiguration," IETF RFC 2462, Dec. 1998; available at http://www.rfc-editor.org/rfc/rfc2462.txt.
6. L. Ebisov et al., "Multicast DNS," Internet draft, DNSEXT WG, Nov. 2000; work in progress.
7. M. Crawford, "IPv6 Node Information Query," Internet draft, IPNGWG WG, Aug. 2000; work in progress.
8. E. Guttman et al., "Service Location Protocol, Version 2," IETF RFC 2608, June 1999; available at http://www.rfc-editor.org/rfc/rfc2608.txt.
9. E. Guttman, "Service Location Protocol Modifications for IPv6," Internet draft, Svrloc WG, Feb. 2001; work in progress.
10. E. Guttman, "The Service Location Protocol," *IEEE Internet Computing*, vol. 3, no. 4, July 2000, pp. 71-80.
11. A. Gulbrandsen et al., "A DNS RR for specifying the location of services (DNS SRV)," IETF RFC 2782, Feb. 2000; available at http://www.rfc-editor.org/rfc/rfc2782.txt.
12. O. Catrina et al., "Zeroconf Multicast Address Configuration Protocol (ZMAAP)," Internet draft, Zeroconf WG, March 2001; work in progress.
13. E. Guttman, "Zero Configuration Networking," *Proc. INET 2000*, Internet Society, Reston, VA; available at http://www.isoc.org/inet2000/cdproceedings/3c/3c_3.htm.

To locate the latest version of an Internet draft, refer to IETF working group information pages at http://www.ietf.org/html.charters/wg-dir.html.

Erik Guttman is a senior staff engineer at Sun Microsystems, living in Germany. His main professional interests are automatic network configuration and practical network security. He is chair of the Service Location Protocol and cochair of the Zero Configuration IETF working groups.

Readers can contact the author at erik.guttman@sun.com.