4-26-2012

The InterMesh Network Architecture-Demo Proposal

Chaouki T. Abdallah

J. Khoury

J. Crichigno

H. Jerez

W. Shu

See next page for additional authors

Authors
Chaouki T. Abdallah, J. Khoury, J. Crichigno, H. Jerez, W. Shu, and G. Heileman

The InterMesh Network Architecture - Demo Proposal

J. Khoury *, J. Crichigno *, H. Jerez†, C. T. Abdallah*, W. Shu *, and G. Heileman *
*School of Electrical and Computer Engineering, University of New Mexico, Albuquerque, NM
{jkhoury, jcrichi, chaouki, shu, heileman}@ece.unm.edu
†Corporation for National Research Initiatives, Reston, VA
hjerez@cnri.reston.va.us

I. Introduction

There is an indispensable need for inter-networking of WPAN-, WLAN-, WMAN- and cellular-based wireless mesh networks. Currently, the Internet protocol (IP) is employed to provide this functionality at the network layer. However, today's Internet implementation, despite its great success, has several limitations, including: 1) the overloading of the IP address to simultaneously indicate network location and node identity, 2) the absence of a trustworthy environment for users to communicate, and 3) the questionable availability of centralized infrastructure and services. The advent of ubiquitous computing paradigms and the success of emerging access networks present an inflection point for introducing fundamental paradigm shifts towards designing a future inter-network.

The InterMesh architecture [1] is a novel architecture with the goal of inter-networking heterogeneous mesh networks to provide a seamless service to individual network entities. The following key design concepts distinguish InterMesh:

- Intrinsic support for unstructured networks;
- persistent identification/naming and certification of network entities;
- a novel approach to dynamic and extensible network management and service provisioning using mobile agents; and
- seamless mobility.

InterMesh achieves convergence through a uniform Persistent Identification and Networking Layer (PINL), allowing mesh communities to form "on-the-fly" and merge with other networks. The PINL identifies network entities with persistent identifiers (PIs), that are globally unique, secure and accountable by design. This paper reviews the system components and functionality, and discusses the proposed conference demonstration.

II. InterMesh System Model and Operation

A high-level sketch of the InterMesh system model on which the demo is directly based, is shown in Figure 1. There are several essential components and constructs within the system, including: *Entity*, *Area of Influence*, *Neutralization Environment*, and *Network Substrate*.

- Entity: Based on the definition in [2], an entity is the end-point of communication. It is an abstract construct that can represent a process, a thread, a device, a cluster of devices, or a service. The entity is the smallest indivisible element on the network that can be mobile. All entities within InterMesh are persistently identified and addressed. For the scope of the demo, an entity will represent a process.

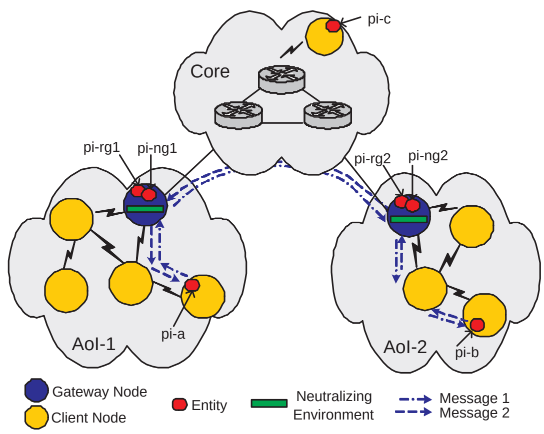

Fig. 1. InterMesh reference model

- Area of Influence - AoI: Whenever we speak of a local network, we are referring to what we call the Area of Influence (AoI). Briefly, an AoI is a local communication community that defines its own communication protocols and network architecture implementations. In this demo, we will use mesh WLAN networks as AoIs.
- Neutralization Environment - The GHOST/SHELL model: Within our system, GHOSTs are necessary to provide efficient and extensible service provisioning and network management. A GHOST is the term we use to represent a mobile agent within InterMesh. A set of nodes within the mesh AoI provides the framework to host GHOSTs. Such a node is referred to as SHELL. Part of those nodes, usually referred to as the gateway nodes, connect the mesh AoI to the "core" substrate (e.g. Internet). The idea is the following: specific nodes (SHELLs) within the AoI expose a neutralizing interface which in essence virtualizes the actual device's hardware resources. GHOSTs make use of this interface to move to a particular device and to implement necessary network services, which include routing and naming services for

the AoI. GHOSTs are themselves entities, hence, they are persistently identified. Figure 1 shows the GHOST entities (*pi-rg1*, *pi-ng1*, *pi-rg2*, *pi-ng2*) executing on top of a SHELL's neutralizing environment. The GHOSTs can move between SHELLs for reasons of resource optimization and fault tolerance. We have specifically addressed the agent relocation problem for the maximization of the network resources as an optimization problem in [3]. Note that the GHOSTs do not represent infrastructural components within the AoI, but on the contrary, they provide dynamic on-the-fly services for the rest of the entities in the AoI. For example, in an emergency (first responder) network, we envision a set of nodes rapidly forming into an AoI with the necessary GHOSTs automatically initializing and relocating to optimize the network utility.

- Network Substrate: In general, the network substrate is composed of the underlying communication infrastructure and services, as well as the mesh nodes. The infrastructure includes the edge mesh networks (AoIs), and a common "core" such as the Internet, the cellular infrastructure, or any other access/distribution network. As to the mesh nodes, we distinguish gateway nodes which specifically serve as entry/exit points between the the mesh network and the "core".

A. Entity Naming

A major design decision adopted in our work is the persistent identification of individual network entities, whereby each entity has its own PI that is globally unique and secure by design. We take an *entity-oriented* approach to naming and communication, in which the entity becomes the first-class network citizen. Contrasted to the traditional IP approach, our starting design point is an entity with a globally unique PI that is independent of any topological information. In order to participate in the system, an entity must acquire a *stamped* PI, i.e., a PI associated with a *stamp*. The latter is a credential acquired from a certification authority (CA) to authenticate the owner(s) of a PI. We have reused a current implementation of the persistent identifier called the *handle* which is part of the Handle System [4]. Briefly, the Handle System provides a distributed, secure and global name service for administration and resolution of *handle*s over the Internet. Entities can thus utilize the InterMesh framework to communicate based on PIs (*handle*s) as network layer identifiers.

B. Protocol Stack

Figure 2.A shows the logical layered structure of the native protocol stack within InterMesh. We have as well ported an overlay version of the stack that operates as an overlay on top of IP networks (Figure 2.B). The ported version is necessary for inter-operability with current IP networks. The lower physical and link layers are common to all mesh networks. The Mesh Structuring Layer (MSL) abstracts the underlying connectivity specifics and routing protocols from the upper layer enabling a wide range of emerging ad-hoc/mesh networks to be part of our architecture. For our implementation, we reuse The Microsoft Mesh Connectivity Layer [5] for this purpose.

The Persistent Identification and Networking layer (PINL) is an inter-networking layer that provides the necessary network services to foster evolution and innovation of the network. Protocols related to mesh network establishment, self-configuration, discovery, and packet delivery between persistently identified entities belong to this layer. Presented with a PI, this layer is intelligent enough to deliver a packet to its destination(s). Briefly, the advantages of using the PI as

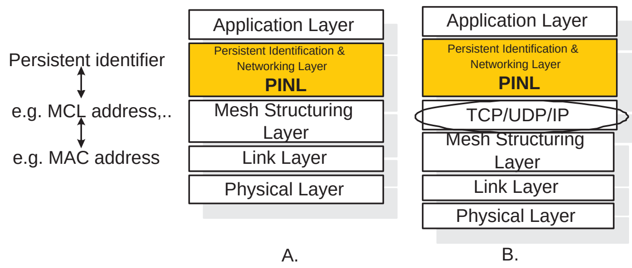

Fig. 2. InterMesh logical protocol stack layering A. Native, B. Overlay

the network address are: 1) *Persistence* - The independence of the PI from its attributes is an attractive property for a network layer identifier. The direct advantage of persistence is mobility since an entity that is persistently addressed by the network layer is reachable on that address at all times, 2) *Security* - The PI address is stamped i.e. it is inherently associated with security information (e.g. public/private keys) which can be used at all times by the communicating parties (and the network if necessary) for accountability, authentication and confidentiality, and 3) *Distributed administration* - The PI has an incorporated administrative model that assigns administrators to PIs with access rights down to the level of the attribute. An administrator of a PI can securely self-administer the PI at anytime, and from anywhere within the network.

The application layer is basically the entity's interface to the network through which the service is provided. This demo will present a simple robot control application at this layer to demonstrate the correct operation of the overall stack.

III. Demo Procedure

Having reviewed the basic components and functionality of the system, we now present the demonstration procedure. The procedure aims at illustrating the main research ideas and their importance. Our demonstration is directly based on Figure 1, where one mesh network will be deployed at the conference site and another remote mesh will be deployed at the ECE department at the University of New Mexico. We show how entities can seamlessly communicate locally and remotely based on PIs.

A. Network Initialization using GHOSTs

We will first demonstrate the GHOST/SHELL model implemented on top of the JINI [6] framework. InterMesh offers naming and routing services by utilizing the GHOST/SHELL model. We will show how the administrator of the AoI

ships the GHOSTs from any point within the network to the particular SHELL. After authenticating itself to the SHELL, a GHOST implements its intelligence independent of the actual SHELL hardware that is abstracted through a neutralizing interface [1]. Two GHOST types will be disseminated into the AoI and these are sufficient to initialize an AoI: *Naming Ghost* - NG, and *Routing Ghost* - RG.

- Naming Ghost: This agent is particularly disseminated into the network with the goal of implementing the identification (name) service for the AoI. It appears as an entity to other entities in the AoI and is identified with a PI. In our model of Figure 1, the naming GHOSTs are represented by entities *pi-ng1* and *pi-ng2* providing the name service for *AoI-1* and *AoI-2* respectively.
- Routing Ghost: The routing GHOST implements the actual overlay routing protocol that delivers packets between the local AoI and the rest of the network. Its operation is similar to the default gateway concept in the traditional Internet except that it routes based on the PI instead of IP address. Figure 1 shows the routing GHOSTs represented by entities *pi-rg1* and *pi-rg2* providing the gateway service for *AoI-1* and *AoI-2* respectively.

Both GHOSTs provide a discovery interface that allows other entities within the network to automatically discover the suitable GHOST.

B. Device and Entity Initialization

The second phase of the demo will show the native stack running on the client nodes within the mesh network. Our PINL layer is implemented over the Windows XP operating system as a combination of user level and kernel level processes. Within this layer, we demonstrate several modules including an NDIS protocol driver, an ARP module, a forwarding-demultiplexing module, and a discovery module. We also show the Microsoft Mesh Connectivity Layer [7] operating below PINL. As the node initializes, we demonstrate how it dynamically discovers and binds to the routing and naming GHOSTs.

After initializing the device, we show how an entity (a process) initializes on the device. The entity obtains/updates its PI and can transparently communicate using the application layer stack API. The entity we will demonstrate is a simple robot control application.

C. Seamless Communication based on persistent identifiers

After initializing the devices and entities, we demonstrate communication between the persistently identified entities (application level processes) locally and remotely. In the former case, two entities on separate devices will interact directly over the local mesh. Note that in this case, a sender node discovers the recipient to be local and communicates directly with the latter over multi-hop with no need for the gateway nodes. In the remote interaction case, the sender node located in the mesh network at the conference site will interact with a recipient located in the remote mesh network at the UNM cite. In both cases, the sender process needs to only know the persistent identifier of the recipient. The InterMesh framework handles the packet delivery which is transparent to the entity.

[1]For example, an upgrade to the AoI routing algorithm simply requires shipping an upgraded GHOST to replace the old routing GHOST.

IV. Additional Demo Information and Requirements

A. Useful Links

An online version of the demo is provided in [8]. Additionally, a technical report describing the InterMesh network Architecture is provided in [1].

B. Requirements

Equipment: We will provide two laptop devices and a gateway device (Mac mini). The laptop devices will demonstrate the native stack and the local communicating parties, while the gateway device will provide the mesh network and demonstrate the operation of the routing and naming agent (GHOSTs). Optionally, we might use a robot to demonstrate the robot control application running on top of the InterMesh network stack as shown in the online demo [8].

Space: A table is enough to deploy the equipment.

Facilities: Internet is required to stream video from a webcam located at the ECE department at UNM. We also need power supply for the devices.

Setup Time: The setup time is around 10 minutes.

C. Note

We would like that you consider this demo for the student demo competition. The lead student is the first author of this paper, Mr. Joud S. Khoury who is a PHD student at the University of New Mexico, ECE department.

V. Acknowledgements

The work reported in this paper is partly supported of the Transient Network Architecture project funded by the National Science Foundation NSF under the Future Internet Design (FIND) Grant CNS-0626380. Additional information can be located at the project's website http://hdl.handle.net/2118/tna.

References

[1] J. Khoury, J. Crichigno, H. Jerez, C. Abdallah, W. Shu, and G. Heileman, "The intermesh network architecture," University of New Mexico, Technical Report EECE-TR-07-007, April 2007, [online]: http://hdl.handle.net/1928/3052.
[2] D. Clark, R. Braden, A. Falk, and V. Pingali, "Fara: reorganizing the addressing architecture," in *FDNA '03: Proceedings of the ACM SIGCOMM workshop on Future directions in network architecture*. New York, NY, USA: ACM Press, 2003, pp. 313–321.
[3] J. Piovesan, C. Abdallah, H. Tanner, H. Jerez, and J. Khoury, "Resource allocation for multi-agent problems in the design of future communication networks," University of New Mexico, Technical Report EECE-TR-07-001, April 2007, [online]: http://hdl.handle.net/1928/2973.
[4] "The handle system," http://www.handle.net.
[5] R. Draves, J. Padhye, and B. Zill, "Routing in multi-radio, multi-hop wireless mesh networks," in *MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking*. New York, NY, USA: ACM Press, 2004, pp. 114–128.
[6] "Jini technology," http://www.jini.org.
[7] "Self-organizing neighborhood wireless mesh networks." [Online]. Available: http://research.microsoft.com/mesh/
[8] "Intermesh online demo," http://hdl.handle.net/2118/intermeshDemo.