ChickWeed: Group Communication for Embedded Devices in Opportunistic Networking Environments

Einar Vollset, Robbert van Renesse and Ken Birman
Cornell University
Ithaca, NY 14850
{einar,ken,rvr}@cs.cornell.edu

Abstract

We present an overview of our group communication system, ChickWeed, for opportunistic networking environments. Opportunistic networking is characterized by sporadic and intermittent connections between wireless devices, such as may arise when embedded devices are carried by humans. ChickWeed is a distributed persistent object store that exploits these connection opportunities through an opportunistic gossiping mechanism. ChickWeed provides the ability for these devices to efficiently and securely share data without requiring access to any infrastructure.

1 Introduction

Embedded systems are frequently used in environments where intermittent and time varying wireless communication opportunities exist. Such opportunities arise when wireless embedded devices are in proximity of each other and are dependent on such unpredictable factors as user mobility, available battery power or local rules (such as needing to turn off electronic devices on airplanes during take-off and landing).

These opportunities are often not the traditional end-to-end connectivity expected at stationary, wired systems such as PCs. Instead, they are connections to devices encountered in our physical environment. Some devices, such as the group of devices owned by a single person, may stay directly connected for relatively long periods of time, while others, such as the devices owned by two distant relatives may only connect briefly or through a number of intermediaries implying no end-to-end path ever exists. Studies of these types of connection opportunities include studies of movement of users on university campuses[4][8] and attendees at conferences[5].

Existing network architectures such as web or email work badly in scenarios lacking access to some form of infrastructure. This is because these applications require more or less permanent end-to-end paths to servers and services, and are structured around streams of data (TCP) to numeric addresses (IP) resolvable through some infrastructure based service (DNS).

Contrary to to traditional end-to-end networked applications, opportunistic networking enables a new style of application that takes advantage of the ad-hoc, intermittent and unpredictable communication opportunities that arise between wireless devices. Due to the inherent spatial and temporal reuse of the wireless medium, such opportunities can co-exist with more traditional wireless communications such as through WiFi access points.

This paper describes the design of a group communication system, called *ChickWeed*, we are building for such an *opportunistic networking* environment. ChickWeed innovates by implementing a novel opportunistic gossiping mechanism that disseminates data by effectively exploiting communication opportunities as they arise.

The gossiping mechanism found in ChickWeed differs from more traditional gossiping mechanism, such as [2], in that our design approach has been to avoid attempting to construct and maintain routing structures. The rationale behind this is that we expect connectivity to be sporadic and intermittent, and that there are potentially huge numbers of devices involved in the system. Both these observations imply that attempting to maintaining routes will incur heavy overheads and be largely ineffective. Instead, whenever two or more nodes are in range of each other they exchange a synopsis of their local state, and reconcile their state as much as possible.

If ChickWeed nodes are equipped with some form of location capability, such as GPS, then the system can take advantage of this by preferentially sending gossip data "towards" destinations. We are not aware of any prior system that has used this kind of physical information in the context of a gossip protocol.

In addition to effectively disseminating data, the ChickWeed system includes mechanisms to ensure the consis-

tency, integrity and confidentiality of data. These requirements naturally arise in the wireless environment ChickWeed is designed for, which has no physical access control and where multiple concurrent updates of data are likely to occur.

The primary application interface to ChickWeed is a filesystem abstraction. This provides a familiar and simple interface with well known semantics to the end user or application. The ChickWeed filesystem interface have been implemented as a Linux user space filesystem.

Such a system can be widely useful, for example through automatic synchronization of groups of user's devices without requiring configuration or access to infrastructure networks. Examples of applications that motivate our work include disaster response, military urban warfare, and team activities such as mountaineering. In all of these cases groups of individuals must cooperate (and would benefit from computing support), despite the lack of infrastructure.

In the next section we describe the structure of our system, including the accompanying filesystem interface. Section 3 briefly reviews related work, while section 4 concludes with future and ongoing work.

2 ChickWeed System Structure

ChickWeed is designed as a distributed persistent object store connected by our opportunistic gossiping mechanism. The main components of the system are *objects* and *updates* to those objects.

An object has a specific type, and application developers can extend ChickWeed by implementing their own objects as needed. Currently two types of objects has been implemented: data objects and directory objects. Data objects contain an array of bytes while directory objects contain a list of strings.

An object consists of two parts: meta-data, and contents. The meta-data contains information about version number, name of object, applied updates, and so on. The version number is a pair consisting of a major and a minor version number. The minor version number is incremented each time the meta-data is updated. If the contents section is updated, the major version number is incremented, and the minor version number is reset to zero.

Object versions are guaranteed to be consistent across multiple nodes; that is, if two nodes have objects with the same version number, then their meta-data and contents are also the same. In order to guarantee such consistency, objects cannot always be updated directly, but have to be updated through a consistency protocol. This is achieved by assigning each object an *authority*. Currently the authority defaults to the node which generated the object. If a node without the authority for an object attempts to update it, an update gets transmitted to the authority, and the authority applies the update.

Updates also have types. When implementing the various objects, application developers specify how to apply a specific update based on its type. For example, a data object will replace all its contents on applying an update of type overwrite, but will append the contents of the update if it receives one of type append.

2.1 Gossip mechanism

The opportunistic gossiping mechanism relies on nodes exchanging synopses or digests of their local state whenever a communication opportunity arises. Upon receiving such a digest, a node compares its local state to that indicated by the digest and transmits any objects or updates it determines the other node is lacking.

Currently the digests of the local state is a (compressed) list of the objects and updates a node has in its local persistent database. Clearly, as the number of elements in the system grows, such an approach will be unsatisfactory, and once we gain experience from testing the running system we expect to refine this mechanism.

We expect previous work on fast approximate set reconciliation[3] and low bandwidth filesystems[9] to prove useful, but we expect to design a novel state exchange protocol to take advantage of the peculiar features of the opportunistic networking environment.

Examples of such features include information about the physical location of devices in the system (through e.g. GPS or similar) and the inherent broadcast nature of the wireless medium.

Gossip mechanisms designed for wired systems typically do not take into consideration the physical location of nodes in the system, as potentially any node is reachable through standard IP routing.

In the opportunistic networking environment however, it may become crucial to introduce directionality or location awareness into the gossip mechanism. The reason is twofold: First, certain data being gossiped may be strongly tied to a specific physical location and be irrelevant elsewhere. Second, the group of intended receivers may be in a specific physical location, and sending the data elsewhere leads to inefficiencies.

The broadcast nature of the wireless medium also presents intriguing opportunities. Traditional protocols are structured around (multiple rounds of) point-to-point data exchange between two nodes at a time. This makes it relatively easy to pull data off, or push data to a specific node in an orderly fashion.

When using broadcast, potentially very many nodes could hear a single request for some data, leading to potentially very many (interfering) responses. However, on the

positive side, a single transmission of data could be useful to potentially very many nodes. We are currently investigating how to best exploit the broadcast nature of wireless in the context of our opportunistic gossiping mechanism.

2.2 Security

In ChickWeed any device which is in wireless range can request and receive objects, so it is crucial that the system incorporates appropriate security mechanisms to provide confidentiality and ensure integrity.

Confidentiality is achieved by encrypting objects based on a per group basis, where a group could for example be all the devices in a given department, owned by a specific person, or simply the devices owned by users in a given meeting.

In order to provide fine grained access control, an object can be part of one or more groups. A group is defined as the nodes which have access to the group's secret key. Currently this key is a symmetric DES key. We assume that the members of the group get given the secret group key prior to interacting with the system (for example projected on a wall in a meeting, or printed in the conference proceedings), and we do not support group re-keying. This may have to be revisited in the future.

The contents of an object is encrypted with a *per-object* symmetric key. The symmetric key is generated when the object is created, and this key is encrypted with each of the shared group keys this object is in. These encrypted versions of the object key is then added to the object meta-data (which is not encrypted). This has the benefit that an object can be in multiple groups without requiring it to be encrypted and transmitted more than once.

In addition to ensuring confidentiality of contents, the system must ensure that contents and meta-data is tamper proof. This is handled by computing and sending a SHA1 digest with each object. On receiving an object, a node first checks if the digest is correct, and if not it ignores it.

2.3 Filesystem interface

The system provides a file system interface as its main application interface for applications that do not need to define their own objects and updates. This has been implemented as a userspace Linux filesystem. Using a filesystem means application programmers and users can easily and cleanly interface with the system.

The mapping between objects and files and directories is straightforward: files are data objects and directories are directory objects. When a user edits a file, these modifications are not shared until the file is closed. These semantics allows for fine grained control of when data is to be shared.

Changes to files that are generated locally are immediately visible through the filesystem, while changes to files the local node did not create are not visible until after the authority of the underlying object has received and applied the update generated.

In addition to the filesystem, a publish-subscribe interface is under consideration for future versions of the system.

3 Related work

Mobile ad-hoc networks (MANETs) have received considerable attention from researchers in the last few years. The main focus in this area has been on attempting to provide end-to-end IP level routing, for example by using protocols such as AODV[10] or DSR[6]. In contrast our approach has been to explicitly avoid the assumption that end-to-end routes can be built at all.

Delay Tolerant Networks (DTNs), such as the Message Ferrying project[12], are focused on networks missing this end-to-end property, and our system is i delay tolerant in this respect. However, the main focus in DTNs has been on longer, more predictable point-to-point links, such as space links (the original DTN vision arose in the IETF Interplanetary Internet Working group) or those formed by sending data by motorcycle courier between villages.

Perhaps the most closely related project to ours is the Haggle Project, introducing the concept of Pocket Switched Networking(PSN)[5]. Haggle is an ambitious attempt to introduce a new network architecture suitable for opportunistic networks. The proposed architecture completely eliminates layering above the data-link, and aims to exploit application-driven message forwarding. The focus so far has been on point-to-point communication, where as in contrast ChickWeed can be thought of as a "group oriented" pocket switched network.

4 Ongoing and Future work

ChickWeed is under active development, and a working prototype that has been tested on the Orbit-Lab [11]wireless testbed is available for download [1].

Our main focus at the moment is to evaluate the effectiveness of various state exchange protocols in the context of opportunistic wireless networks. An interesting avenue is how the broadcast nature inherent in wireless networks impacts such protocols, as most proposed protocols so far has focused on reconciliation between two well connected peers.

Future work includes increasing fault-tolerance and availability by introducing multiple authority objects. Such an addition would require implementing an agreement protocol, such as Paxos[7], to maintain consistency. How

agreement protocols perform in real wireless networks is also an open question.

Finally, we're looking at how to scale our system in number of participating nodes, and whether geographic information, such as that provided by GPS is helpful in this respect.

References

[1] http://www.cs.cornell.edu/ einar/chickweed/, August 2006.

[2] K. P. Birman, M. Hayden, O. Ozkasap, Z. Xiao, M. Budiu, and Y. Minsky. Bimodal multicast. *ACM Transactions on Computer Systems*, 17(2):41–88, 1999.

[3] J. Byers, J. Considine, and M. Mitzenmacher. Fast approximate reconciliation of set differences. Technical Report 2002-019, Boston University, 2002.

[4] T. Henderson, D. Kotz, and I. Abyzov. The changing usage of a mature campus-wide wireless network. Technical Report TR2004-496, Dartmouth College, 2004.

[5] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot. Pocket switched networks and human mobility in conference environments. In *WDTN '05: Proceeding of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 244–251, New York, NY, USA, 2005. ACM Press.

[6] D. B. Johnson and D. A. Maltz. Dynamic source routing in ad hoc wireless networks. In Imielinski and Korth, editors, *Mobile Computing*, volume 353. Kluwer Academic Publishers, 1996.

[7] L. Lamport. The part-time parliament. *ACM Transactions on Computer Systems*, 16(2):133–169, 1998.

[8] M. McNett and G. Voelker. Access and mobility of wireless PDA users. In *Mobile Computing Communications Review*, volume 9, pages 40–55, 2005.

[9] A. Muthitacharoen, B. Chen, and D. Mazieres. A low-bandwidth network file system. In *Symposium on Operating Systems Principles*, pages 174–187, 2001.

[10] C. Perkins. Ad-hoc on-demand distance vector routing. In *MILCOM*, 1997.

[11] D. Raychaudhuri, I. Seskar, M. Ott, S. Ganu, K. Ramachandran, H. Kremo, R. Siracusa, H. Liu, and M. Singh. Overview of the orbit radio grid testbed for evaluation of next-generation wireless network protocols. In *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, 2005.

[12] W. Zhao, M. Ammar, and E. Zegura. A message ferrying approach for data delivery in sparse mobile ad hoc networks,. In *Proc. of the 5th ACM international symposium on Mobile ad hoc networking and computing (Mobihoc)*, pages 187–198, 2004.