CASCADE: Content Access System for the Combat-Agile Distributed Environment

Tim Strayer, Vikas Kawadia, Armando Caro, Samuel Nelson, Dorene Ryder
Raytheon BBN Technologies
Cambridge, MA, USA
strayer|vkawadia|acaro|snelson|dryder
@bbn.com

Carsten Clark, Kolia Sadeghi
Commonwealth Computer Research, Inc (CCRi)
Charlottesville, VA, USA
carsten|kolia@ccri.com

Bryan Tedesco, Olivia DeRosa
Future Skies, Inc
Wall Township, NJ, USA
Bryan.Tedesco|Olivia.DeRosa
@future-skies.com

***Abstract*—Military dismounted communications are critical to the success of missions and safety of soldiers, and hence are strongly regimented and hierarchical. Certain situational awareness apps have been introduced, where content is exchanged between soldiers in a squad, but soldiers primarily carry any needed content out with them and bring any collected content back to base. Furthermore, soldiers rarely share content among themselves or with other squads, and any readily needed content must be retrieved through an expensive "reachback" link to base. We present CASCADE, a content-centric networking architecture that facilitates generation and dissemination of content based on soldiers' interests, both statically assigned (e.g., specialty within the squad) and dynamically discovered (i.e., derived from context). The key enabling abstraction within a CASCADE network is the concept of communities. Through the use of both topological and interest-based communities, CASCADE is able to quickly and efficiently deliver content, even in environments prone to disruption.**

Keywords—content-centric networking; information-centric networking; MANET; disruption tolerant networking

I. INTRODUCTION

The success of ground operations conducted by units at the tactical edge depends heavily on access to relevant and timely information, including social data, biometric data, threat assessments, geospatial imagery and fused intelligence, all from multiple data sources and services typically hosted on nodes within a mobile ad hoc network (MANET). The nodes sourcing and requesting this data are subject to bandwidth, latency, and connectivity constraints that have the potential to severely degrade the accessibility of crucial content. Tactical networks present a challenge to the efficient distribution of application data to MANET nodes and to the availability of data in imperfect, real-world network conditions.

Current network services at the tactical edge suffer from characteristics inherited from the Internet. First, addressing is static and flat, and the address has no relationship with the content that is being transported. Second, any association between those who need certain types of content and those who produce it must be established *a priori*; the network plays no role in making this association. Third, the content caching technology that has greatly improved the responsiveness and efficiency of the Internet cannot easily be applied within the dynamic topologies of tactical edge networks. Without the benefit of caching, Internet-motivated designs make current tactical networks unresponsive and fragile.

The paradigm of content-based networking offers a solution to these problems. Every node is a peer and there is no central point of access for any given content object. This simplifies caching by decoupling it from topological dynamism. A content-based network provides a decentralized service that can be used to build robust and reliable applications by leveraging the redundant usage of content among network users. This paradigm offers a new way of viewing the digital world, one where content objects, and not end hosts, are the fundamental routing elements. Networking operations change: instead of asking for a connection to a foreign host, higher layer protocols ask for access to a particular piece of content or a description that can resolve to content. A content-centric network provides the abilities to publish, query for, and describe content as primitive operations. Since tactical apps, such as TIGR[6] and Nett Warrior [7] care about content and not hosts, this paradigm fits well into the tactical edge environment.

Applying the content-based networking paradigm to tactical MANETs poses a unique set of challenges, including rapidly and unpredictably varying links, constrained computing and networking resources, mobility, and potential network partitioning. In response, we present CASCADE, a content-based networking architecture specifically designed to overcome the challenges associated with tactical MANETs. By the intelligent replication and placement of content, CASCADE is able to quickly serve relevant content to nodes in the MANET while operating in a resource-friendly fashion. Furthermore, CASCADE provides a rich querying interface to quickly and intuitively discover content.

The goal of CASCADE is to balance the latency incurred by content queries with the overhead incurred by moving and replicating the content itself, while maintaining delay-tolerant access to content. The underlying philosophy that allows CASCADE to meet this goal is to apply a hybrid "push-pull" approach, where content is proactively *pushed* to strategic

locations such that it can be *pulled* from those locations on-demand quickly and efficiently. CASCADE dynamically manages this fine balance through the use of *communities*, which extract both topological and contextual information from the network and the soldiers.

In the rest of this paper, we present an overview of the hybrid push-pull philosophy, present our community-based mechanisms for realizing this philosophy, and show performance benefits via an Android implementation over an emulated network.

II. CASCADE DESIGN AND ARCHITECTURE

At the heart of CASCADE is the use of *communities* to help regulate how content is pushed and pulled from stategic locations in the network. A community is a set of nodes bound together by some property. CASCADE currently focuses on two properties – location and mission-relevant context. The resulting types of communities are referred to as *topological communities* and *interest-based communities*. Topological communities assist in the decisions to proactively place content in topologically stable regions of the network, such that nodes in those regions can quickly access this content. Interest-based communities assist in the decisions to place content close to nodes that are likely to access the content, as determined by their context. CASCADE's algorithms utilize both of these types of communities to meet its latency, overhead, and disruption-tolerance goals.

The CASCADE architecture (see Figure 1) is highly modularized by design, with each module serving a well-defined purpose. There are five primary modules: (1) Content-Distribution System (CDS), (2) Content Management System (CMS), (3) Content Storage System (CSS), (4) Content Naming System (CNS), and (5) the Mediator. The CDS controls the propagation and routing of content and queries, and is the module that utilizes topological communities. The CMS provides intellegence regarding which nodes are interested in which content, and is the module that utilizes interest-based communities. The CSS is the storage/caching system for each node. The CNS is the content discovery system for each node. The Mediator is a shim that allows existing, non-CASCADE-aware apps to talk over a CASCADE network.

CASCADE has been implemented on the Android platform. Our discussion here is focused more on the architecture and design, and less on implementation details due to space constraints.

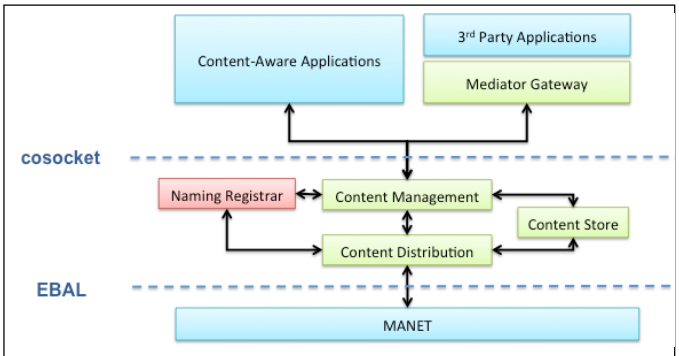

Figure 1: CASCADE Architecture.

A. CASCADE Modules

The CDS deals with network placement and retrieval of uniquely named data objects (NDOs). These objects may be content such as pictures, documents, or videos, or they may be metadata containing information about other objects. CDS is agnostic to what is inside an object; it only requires that objects have persistent, location-independent, globally unique names that allows CDS to intelligently place them in the network and efficiently retreive them later. CDS thus has a similar objective to information centric networks such as CCN [3], which are designed for Internet scale wired networks. CASCADE however is focused on tactical MANETs and goes beyond opaque data objects to richly named content and metadata and their management based on context and relavance.

The CMS is responsible for ensuring that content most relevant to the user is highly available when needed. CMS provides this functionality by using historical access patterns to estimate the access rates of documents within different contexts and by using those estimates to influence caching decisions. CMS may proactively pre-place content within a topological region of the network, and it may opportunistically cache relevant content in coordination with CDS as content flows along a network path.

The CSS is a common data store used by all components on a CASCADE node. CSS stores only one copy of content that is used by both CMS and CDS, thus reducing the total storage space used by the CASCADE system. CSS also implements a storage policy (guided by inputs from CMS and CDS) to ensure that the total storage used is within allowed limits.

The CNS, developed by our partners at Drexel University, provides a rich naming capability that is built on the unique names provided by CDS. The CNS translates queries from the powerful SPARQL [9] language into specific content IDs based on metadata associated with content. CASCADE does not send rich queries out into the network. Instead, a rich query is resolved locally into unique object names, which are then retrieved from the network, by the CDS in an efficient manner.

CASCADE requires few underlying networking abstractions, namely neighbor discovery, reliable 1-hop unicast, and broadcast ability. Furthermore, CASCADE requires an interface to speak with apps, allowing query and publish requests to enter the system. In this regard, CASCADE has partnered with University of Southern California/Information Sciences Institute (USC/ISI), who have developed a software package called COMET, providing these features in the forms of EBAL (Enhanced Broadcast Abstraction Layer) and "cosocket".

B. Native and Legacy Applications

The Mediator's responsibility is to enable communications between third party applications that are not content-aware, with CASCADE's content-aware technology, with zero to minimal alteration on the application side. The Mediator takes advantage of the Android architecture, which provides multiple

paths for exchanging and sharing information, to provide this functionality.

The Mediator consists of Mediator Applications (MAs) and the Mediator Engine (ME). MAs are created for each third party application, with the specific implementation dependent on the Android data exchange path used for communication. MAs provide a set of instructions that define how the ME should handle content from the third party application. A software development kit (SDK) is provided and examples to enable third parties to develop MAs for their Android applications. The SDK also provides a means to communicate directly with the ME using Android Intents in the event that an MA cannot be created for a third party application.

The ME is responsible for 1) receiving any information (content, metadata, queries or notifications) made available by the third party application, 2) interacting with the metadata generation component, 3) translating content and queries (if necessary), and 4) forwarding the information to the CASCADE system. The ME also receives query responses, content updates and notifications from CASCADE and forwards them to the appropriate application based on the MA-provided instructions.

III. CASCADE ALGORITHMS AND PERFORMANCE RESULTS

A. Content Distribution via Topological Communities

The primary objective of the CDS is to efficiently publish content into the network and retrieve requested content from the network. To accomplish this, the CDS algorithms are grounded in the science of topological community detection in networks. Communities are groupings of nodes such that intra-community paths are much more stable than inter-community paths [1]. Topological communities are the mechanism used by CDS to dynamically balance the latency/overhead tradeoff. CDS uses a discovered topological community as distributed storage, such that only one copy of a piece of content is needed within the community to efficiently serve that content to the entire community. In other words, nodes within a topological communuity pool their storage resources together to cooperatively store content without unneeded redundancy. CDS uses distributed hash tables (DHT [4]) to realize such a cooperative storage.

The operation of the CDS can be broken into three components: (1) community formation, (2) community utilization, and (3) inter-community content exchange.

1) Community Formation

CDS actively forms and maintains communities by discovering topologically stable regions within the network. Both spatial and temporal aspects of stability are considered when forming communities. Each node participates in the discovery and formation of a topological community of nodes that share stable connectivity. Every node listens to the control messages of neighbor nodes and upon seeing enough nodes with stable connectivity, announces itself as an anchor for the community. Other nodes that also have stable connectivity to the anchor then join that anchor's community. The community grows as more nodes in the stable region add themselves to the community until the community reaches a certain size or until there are no more nodes within the stable region.

CDS also accounts for the community formation process to be modulated with additional information about military hierarchy, if available. For example, we can stipulate that the communities follow the squad structure --- that is, nodes could join a community only if they belong the squad representing that community. This is an important feature in community formation because it allows the overlay of military structures on the network. It also allows squads to meet each other without thrashing in and out of a larger community; the community structure is maintained, even as the content is exchanged between them.

The CDS has two main community detection algorithms, anchor based and label-propagation based. The anchor-based algorithm is a randomized and distributed implementation of an algorithm to find *r*-dominating sets in networks. The idea is to select a subset of nodes the can be used to define regions of stable connectivity in the MANETs in a completely distributed fashion. This algorithm is based on Slinky [5], and we refer the reader to that work for more details. The label-propagation approach works by propagating labels (or node IDs) periodically, such that nodes change their label based on the majority label that they overhear. This approach is adapted for mobile environments through the use of trigger reset messages that cause all nodes within a community to reset their labels in the event of significant topological change.

2) Community Utilization

Community utilization comprises the way CASCADE exploits the community structure in order to increase availability of content while reducing the latency for acquiring content. Since the definition of a topological community is that it is a collection of nodes that share stable connectivity, CDS can take advantage of this stability by caching one copy of content within the community.

Once communities are detected, nodes run a link state protocol limited in scope to their community. This allows them to discover routes to all other nodes within the community. In order to facilitate robustness, a single node does not serve the entire community; rather, all nodes within the community cooperatively pool their storage resources together via consistent hashing [2] which provides tolerance to churn, i.e., changes in the membership of a community. Specifically, both content as well as the current set of node IDs within the community are hashed to a logical space, and a distance metric (such as the XOR) is used to determine which node is closest to that community in the logical space. This *responsible node* is then given the content via standard shortest-path routing.

Since content is stored within each community, no global topology or server locations are needed; publishes and queries are satisfied by simply contacting the computed responsible node. Also, since communities are, by definition, topologically stable regions of the network, proactively pushing content to communities allows for low latency retrieval and high availability in the face of disruptions in network connectivity.

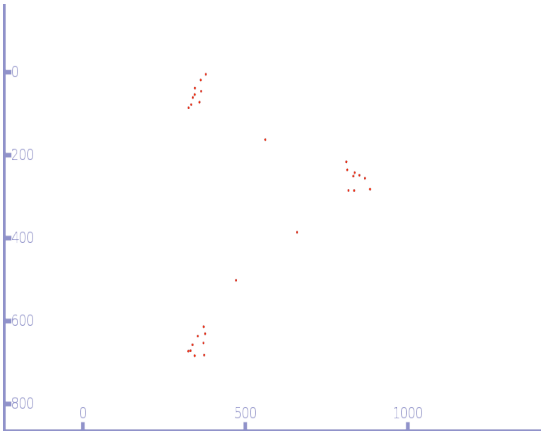

Figure 2: Representative snapshot of the topology with roaming nodes used for evaluations.

3) Inter-community Content Exchange

After content is transferred to the responsible node of the publishing community, it must then be proactively pushed to all other interested communities in the network. The most primative form of the CDS attempts to push a single copy of all content to all communities. The responsible node computes a set of *relay nodes*, or nodes that are on the outer edge of the community with possible connections to other adjacent communities. These relay nodes transfer the content to the adjacent communities, who repeat the process. Furthermore, when two communities become adjacent (due to, perhaps, a partition healing), all nodes pairwise sync their content with the corresponding responsible node. Note that, since content is stored only once in each community, each piece of content crosses the community boundary only once during a sync.

Some communities, however, may not be interested in all content, as defined by their *interests*. The CDS works with the CMS during a community sync to prioritize and send only the most relevant content to adjacent communities. More information on the CMS is in the following subsection.

Note that some queries cannot be satisfied by local content, but rather must either fail, acquire the content by reachback to the base, or wait until content is published locally. The CDS seamlessly interacts with centralized servers at base via means of a *reachback client* that lives on selected nodes in a community. This client facilitates query and subsequent publish operations of content into and out of the CASCADE network.

4) Performance evaluation

To highlight the benefits of the CDS hybrid push/pull approach by means of topological communities, we present a performance evaluation that illustrates the fundamental tradeoffs between latency and overhead. CASCADE is compared against epidemic routing (pure-push) and a centralized global cache (pure-pull). CASCADE is implemented in Android, and each node in the evaluation is a full Android emulator running CASCADE. The underlying network is simulated in ns3, connected to the emulators using Linux tun/tap devices. For the evaluations, we use a tactically

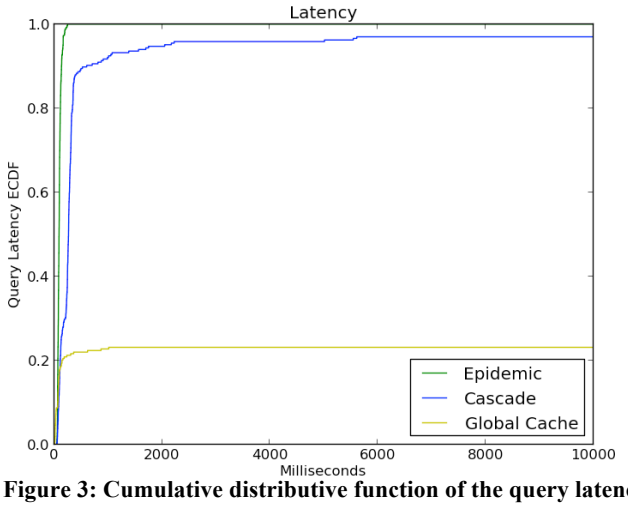

Figure 3: Cumulative distributive function of the query latency shows that CASCADE performs almost as well as the pure-push Epidemic protocol.

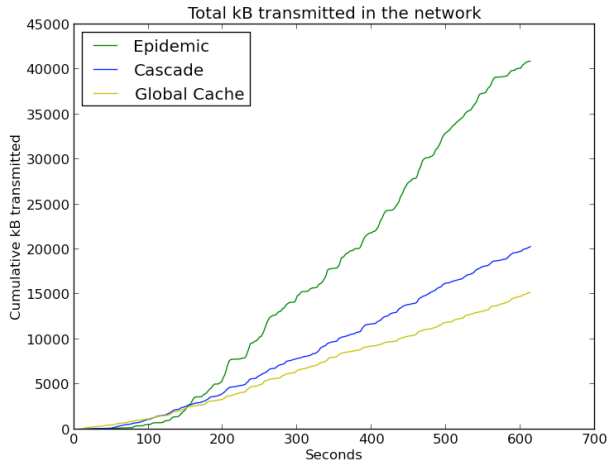

Figure 4: The network overhead of CASCADE is almost as low as the pure-pull Global Cache approach.

relevant topology with three highly affiliated and partitioned squads of nine nodes each, and three roaming nodes moving between squads (see Figure 2). There is some random movement within a squad. Every node publishes content once every 3-4 minutes and queries for content once every minute.

As seen in Figure 3, CASCADE is able to obtain latencies comparable to epidemic, while keeping overhead almost as low as the centralized server approach (see Figure 4). This illustrates how CASCADE uses topological communities to "get the best of both worlds". We also emphasize the very high availability of CASCADE compared to the global cache approach.

B. Content Management via Interest-based Communities

The focus of CASCADE's Content Management Service (CMS) is to include relevance-based prioritization in cache replacement policies, content exchanges, and content pre-placement. CMS does this by modeling content access patterns

as a function of context, exchanging and aggregating models among nearby nodes, and applying those models to prioritization operations.

Content relevance models are computed by observing user access patterns within a given context. Context can include a wide range of parameters, some of which may be provided by the user or application (e.g., user role and operation) and others directly obtainable on the system (e.g., geographical location and time of day). Access patterns consist of application interactions with the CASCADE system, including semantic-based queries and requests for specific content items. By design, CMS is central to the CASCADE architecture to give it the ability to interact with the application and be involved in the high-level internal operations required to fulfill queries and requests.

The relevance models are based on Very Fast Decision Tree (VFDT) [10] incremental regression trees. These have minimal resource requirements, both when they are used to compute relevance, and when model parameters are learned from data, making them ideal for mobile computing. Relevance models may, but need not, be computed, synced, and distributed in real-time; whether or not they are computed offline is orthogonal to the algorithms that use them. If model learning is done online, relevance estimates become gradually more precise as CASCADE is used over time.

Given a node's preference model and context, CMS can compute the relevance score of a content item. The relevance score represents the rate at which the node(s) are expected to access the content item, given that the node(s) are in the provided context. CMS uses relevance scores to prioritize content in several operations.

As a safety measure against potential connectivity issues in a MANET, nodes would opportunistically cache *all* content if storage resources were unlimited. But storage limitations are a reality in theater, making smart replication and caching mechanisms critical for performance and scalability. CMS's cache replacement policy allows a node to opportunistically cache any content it encounters, without much concern for filling up the local cache. The least relevant content is replaced first, thereby offering the most utility to the local and nearby nodes.

CMS also helps CDS prioritize content exchanges among nodes. CDS determines which content should be exchanged given its constraints and needs, and then provides that set to CMS for prioritization. The motivation is to effectively use the bandwidth available by ensuring that the most relevant content is exchanged in the event that connectivity is lost.

Finally, CMS uses relevance to pre-place content near nodes that are expected to access them. Content Diffusion is CMS's algorithm for pre-placing content.

1) Content Diffusion

CASCADE diffuses content throughout the network such that content is placed close to nodes for which it is relevant. The motivation is to amortize the pre-placement costs with expected future accesses. If content is highly relevant in a particular region of the network, then that content is expected to be accessed frequently in the near future, and more effort may be spent pre-placing the content near that region. CASCADE aims at balancing the cost of proactive placement with the content's relevance to the nodes in the vicinity of its placement. Since content relevance is directly computable from a node's context, nodes can reason about how much work should be spent moving a given content item closer to a node.

Content Diffusion parallels distance vector routing in that control information is exchanged among peers to form a common operating state from which nodes make independent decisions for proactively transmitting content. In Content Diffusion, the control information includes the minimum resource cost (e.g., number of hops) to reach a given network region (node or set of nodes), a representation of that region's context, and the set of content already possessed within that region. Each node uses that control information to decide which content should move closer to which neighboring region. Proactive content movements are prioritized at each hop according to relevance and cost to effectively use bandwidth and storage resources.

Although Content Diffusion may operate on a node-by-node basis, CASCADE utilizes *Inter-Community Diffusion*, where CMS interacts with CDS to diffuse content at the granularity of topological communities. In other words, CASCADE aggregates node contexts within each topological community, reasons about content relevance with respect to topological communities, and diffuses content across topological communities (instead of nodes). This approach leverages the identified stable regions of the network to cooperatively store proactively pre-placed content that is relevant to nodes in that region. In essence, Inter-Community Diffusion provides a mechanism to identify and layer interest communities over topological communities to efficiently make relevant content readily available.

2) Performance Evaluation

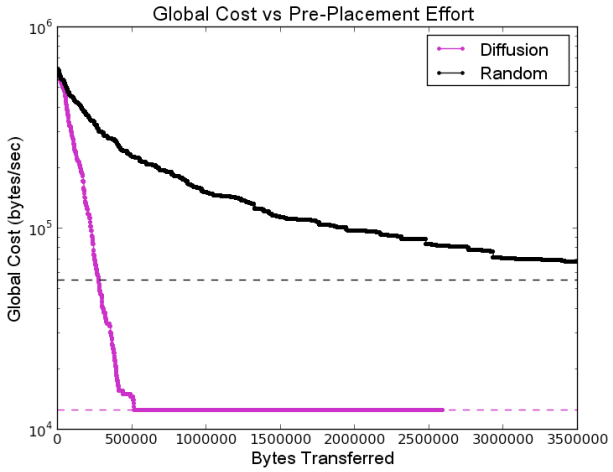

Figure 5: Content Diffusion brings the aggregate cost to the theoretical minimum with much less pre-placement effort than the random content movement scheme.

Figure 5 shows the result of a 12-node simulation where Content Diffusion was compared against random content movement. Nodes published three varieties of content at fixed intervals. Each variety had fixed relevance to a different subset

of the 12 nodes. The experiment measured aggregate cost, which is the sum over all nodes of the distance from a node to each document that is relevant to the node. This figure plots aggregate cost against cumulative bytes moved using either Content Diffusion or random content movement. Content Diffusion brings the aggregate cost to the theoretical minimum with much less effort than random movement.

IV. APPLICATIONS

We have written Mediator shims to allow legacy applications to work on CASCADE. Our experience shows that the Mediator is a powerful abstraction that allows legacy applications to easily work on CASCADE. In many cases, no changes are required in the legacy application. We describe our experience in writing mediators for two legacy applications.

A. Handling Legacy Applications

Dismounted Ad hoc Reporting Tool (DART) is an application that provides a set of reporting options in the field. The app provides VMF XML message and person of interest (POI) photo creation, publish and query. Query for VMF messages resembles the Data Dissemination Service (DDS) format, using upper and lower bounds for latitude and longitude, category, type, etc. POI query has the ability to query based on keywords, within a bounding box area or both. All interactions of DART are integrated with CASCADE through the Mediator using a Mediator Application with the Content Observer communication path.

Traq is an application developed by PARC that generates and stores data for mission analysis. Along with collecting GPS location during movement, the application can capture photo, video and audio messages and store them on the phone. The application was initially created to be standalone with no network or inter-application connection. The Mediator transforms Traq into a networked application that can communicate it's content with other CASCADE-aware applications. At this time Traq does not provide a query function, therefore the Mediator uses the subscribe functionality in order to publish relevant files to Traq that it receives from other CASCADE enabled applications. The Mediator Application created for Traq configures this subscription.

V. CONCLUSIONS

CASCADE provides quick access to content at the tactical edge at low network overhead even in the presence of disruptions. It also provides a rich querying capability to build novel powerful applications. It also allows legacy apps to run unmodified and for applications to share content among each other. Our design and implementation has been tested on networks of up to 30 nodes. We are currently focused on the challenge of scaling up to more nodes and larger and more numerous pieces of content. In addition to the community based dissemination and storage, efficient use of the wireless broadcast medium is key to achieving scalable performance.

ACKNOWLEDGMENT

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) and SPAWAR Systems Center Pacific (SSC Pacific) under Contract No. N66001-12-C-4050. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of DARPA or SSC Pacific.

Distribution Statement "A": Approved for Public Release, Distribution Unlimited, Case 21220.

REFERENCES

[1] S. Fortunato, "Community Detection in Graphs," *Physics Reports*, Volume 486, Issues 3-5, February 2010, Pages 75-174, ISSN 0370-1573.

[2] D. L. Karger. (1997). Consistent hashing and random trees: distributed replication protocols for relieving hot spots on the World Wide Web. Proceedings of the twenty-ninth annual ACM symposium on Theory of computing , 654-663.

[3] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. Briggs and R. Braynard. "Networking named content," *Proceedings of the 5th ACM International Conference on Emerging Networking Experiments and Technologies (CoNEXT 2009)*, 2009 December 1-4; Rome, Italy. NY: ACM; 2009; 1-12.

[4] H. Balakrishnan, M. F. Kaashoek, D. R. Karger, R. Morris and I. Stoica. (2003). Looking up data in P2P systems. *Commun. ACM , 46* (2), 43-48.

[5] V. Kawadia, N. Riga, J. Opper and D. Sampath. "Slinky: An Adaptive Protocol for Content Access in Disruption-Tolerant Ad Hoc Networks," *International workshop on Tactical MANETs in conjunction with ACM MOBIHOC*, May 2011, Paris, France.

[6] TIGR: Tactical Ground Reporting System, General Dynamics, http://www.gdc4s.com/Documents/Programs/TIGR%20Handout-Final.pdf

[7] Nett Warrior, PM SWAR https://peosoldier.army.mil/docs/pmswar/Nett-Warrior-Poster-061512.pdf

[8] Y. Xue and D.E. Brown, "Spatial Analysis with Preference Specification for Latent Decision Makers for Criminal Event Prediction," *Decision Support Systems*, Volume 41, Issue 3, March 2006, pp. 560-573.

[9] W3C, "SPARQL Query Language for RDF," W3C Recommendation, 15 January 2008.

[10] P. Domingos and G. Hulten. "Mining High-Speed Data Streams," *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2000.