AFRL-IF-RS-TR-2005-87
Final Technical Report
March 2005

ADAPTIVE PROBABILISTIC PROTOCOLS FOR ADVANCED NETWORKS/ASSURING THE INTEGRITY OF HIGHLY DECENTRALIZED COMMUNICATIONS SYSTEMS

Cornell University

Sponsored by
Defense Advanced Research Projects Agency
DARPA Order No. H533

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

STINFO FINAL REPORT

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2005-87 has been reviewed and is approved for publication

APPROVED: /s/
PATRICK M. HURLEY
Project Engineer

FOR THE DIRECTOR: /s/
WARREN H. DEBANY, JR.
Technical Advisor
Information Grid Division
Information Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503

1. AGENCY USE ONLY (Leave blank)	**2. REPORT DATE** March 2005	**3. REPORT TYPE AND DATES COVERED** **Final** **Jun 99 – Jun 04**

4. TITLE AND SUBTITLE

ADAPTIVE PROBABILISTIC PROTOCOLS FOR ADVANCED NETWORKS/ASSURING THE INTEGRITY OF HIGHLY DECENTRALIZED COMMUNICATIONS SYSTEMS

5. FUNDING NUMBERS

G	- F30602-99-1-0532
PE	- 62301E
PR	- H533
TA	- 10
WU	- 01

6. AUTHOR(S)

Kenneth Birman
Robert Constable

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Cornell University
4130 Upson Hall
Ithaca NY 14853-7501

8. PERFORMING ORGANIZATION REPORT NUMBER

N/A

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Defense Advanced Research Projects Agency
3701 North Fairfax Drive
Arlington VA 22203-1714

AFRL/IFGA
525 Brooks Road
Rome NY 13441-4505

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

AFRL-IF-RS-TR-2005-87

11. SUPPLEMENTARY NOTES

AFRL Project Engineer: Patrick M. Hurley/IFGA/(315) 330-3624 Patrick.Hurley@rl.af.mil

12a. DISTRIBUTION / AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 Words)***

The project consisted of two merged DARPA-funded efforts exploring related but different topics.

The first was a project to develop a new kind of probabilistically scalable, stable, communications protocols and to exploit these protocols in building a new kind of scalable software infrastructure for large, dynamic, mission-critical networked applications. Examples include networks exploited to support massive data centers (such as are used by the NSA and CIA, by a great variety of military applications, and by a new generation of lightweight sensor networks).

The second was a project to elaborate on a new “compositional” method for protocol design and implementation, in which small microprotocols are combined to obtain a protocol customized to the needs of a specific setting, under control of an automated theorem proving system that can guarantee correctness of the resulting specialized protocol, subject to the validity of assumptions that guide the process. The second was a project to elaborate on a new “compositional” method for protocol design and implementation, in which small microprotocols are combined to obtain a protocol customized to the needs of a specific setting, under control of an automated theorem proving system that can guarantee correctness of the resulting specialized protocol, subject to the validity of assumptions that guide the process.

14. SUBJECT TERMS

Scalable, reliable, secure, adaptive, self-repairing, epidemic protocols, peer-to-peer, gossip protocols, formal theorem proving

15. NUMBER OF PAGES 56

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

ABSTRACT

Over a period of five years, Cornell University was funded by DARPA/AFRL to explore long-term challenges and technology issues in the areas of fault-tolerant networks, protocol guarantees, properties and correctness, security and scalability. Although this is a broad agenda, the work is unified by a single vision: the creation of technologies that will be needed to ensure the robustness, security and scalability of DoD Global Information Grid (GIG) and Network Centric Enterprise (NCES) visions. This final report documents a wide range of contributions and technology transitions, including:

- A new means of matching the "quality of service" for a distributed communications protocol to the requirements of the application and the properties of the environment. The basic idea is to view a communications protocol as offering a set of guarantees: such and such a security property, such and such a consistency property, etc. Each guarantee corresponds to a thin protocol layer – a "microprotocol" and the overall properties of a communications subsystem are obtained by composition of multiple such microprotocols to obtain a protocol stack that offers the combined properties. By picking the stack to match the needs of the application the system pays only for properties it actually requires.

- Record-setting performance in Horus and Ensemble, two popular packages developed by us using this new layered methodology. This work has transitioned into the IBM WebSphere product (the "DCS" subsystem, in WebSphere version 6.0). IBM's product is a market leader and will be widely used by GIG and NCES developers for decades into the future.

- Foundational research on the theory and practice of designing high assurance protocols that can be verified mechanically, a methodology that slashes the risk of human error and can also improve programmer productivity.

- A new approach to building scalable systems using protocols adapted from the peer-to-peer computing arena, and several important proofs of the concept that have already transitioned into major industry cluster-computing products and platforms. In particular, Microsoft's Windows Clustering system, in the Longhorn release, employs this technology.

Our work also entails assisting the military and other government agencies in defining new research programs and architecting their GIG systems. We've worked closely with the Air Force Joint Battlespace Infosphere (JBI) team and have developed productive collaborations with military vendors that range from IBM and Microsoft (mentioned above) to Raytheon, Lockheed Martin and Telcordia. We helped the AFRL JBI team brief the AF SAB on scalability

challenges and related QoS issues in the JBI, and our ideas are prominent in the thinking for JBI's Prometheus design and build. Moreover, DARPA created two funding programs to explore consequences of some of our work: The IPTO Self-Regenerative Systems (SRS) program, under Lee Badger, builds on work done in our effort, and the IPTO Situation-Aware Protocols In Edge Network Technologies (SAPIENT) program, under Jonathan Smith, seeks to take next steps in the area of compositional protocols, inspired in part by our accomplishments. We've also maintained an active dialog and visibility within industry standards groups, such as the W3 consortium (developing Web Services standards) and the Autonomic Computing Consortium, exploring self-management and self-repair options for industry platforms.

Table of Contents

List of Figures

INTRODUCTION

This is the final technical report for three related research efforts conducted at Cornell University over a multi-year period:

- Cornell's Spinglass Project, funded by DARPA ITO in the Fault Tolerant Networks Program under Doug Maughan, and administered through AFRL (Rome) under Patrick Hurley. This consisted of two sub-efforts
    - A project to develop new kinds of probabilistically scalable, stable, communications protocols and to exploit these protocols in building a new kind of scalable software infrastructure for large, dynamic, mission-critical networked applications. Examples include networks exploited to support massive data centers (such as are used by the NSA and CIA, by a great variety of military applications, and by a new generation of lightweight sensor networks).
    - A project to elaborate a new "compositional" method for protocol design and implementation, in which small microprotocols are combined to obtain a protocol customized to the needs of a specific setting, under control of an automated theorem proving system that can guarantee correctness of the resulting specialized protocol, subject to the validity of assumptions that guide the process.
- A seedling project for the new DARPA Self-Regenerative Systems (SRS) effort. This seedling explored the possibility of selecting a set of "best of breed" technology base for building a new kind of distributed platform in support of a wide range of applications that must operate extremely reliably and exhibit self-(re)configuration, self-diagnosis and self-repair capabilities while tolerating failures and maintaining security.

Although we considered drafting a two-part report, we found that doing so resulted in a complex and confusing document and as such would not satisfy the recommendations AFRL has asked us to follow in developing this report. Accordingly, our presentation follows a more or less chronological progression, reviewing first the work undertaken in the two sub-efforts funded under the DARPA FTN program. Then we discuss the follow-on activity that was conducted under the seedling funding, leading to our architecture and preliminary work on QuickSilver.

Broadly, the Cornell University work explores long-term challenges and technology issues in the areas of fault-tolerant networks, protocol guarantees, properties and correctness, security and scalability. This agenda is unified by a single vision: the creation of technologies that will be needed to ensure the robustness, security and scalability of DoD Global Information Grid (GIG) and Network Centric Enterprise (NCES) visions. As noted in the abstract, major results include:

- A new means of matching the “quality of service” for a distributed communications protocol to the requirements of the application and the properties of the environment. The basic idea is to view a communications protocol as offering a set of guarantees: such and such a security property, such and such a consistency property, etc. Each guarantee corresponds to a thin protocol layer – a “microprotocol” and the overall properties of a communications subsystem are obtained by composition of multiple such microprotocols to obtain a protocol stack that offers the combined properties. By picking the stack to match the needs of the application the system pays only for properties it actually requires.

- Record-setting performance in Horus and Ensemble, two popular packages developed by us using this new layered methodology. This work has transitioned into the IBM WebSphere product (the “DCS” subsystem, in WebSphere version 6.0). IBM’s product is a market leader and will be widely used by GIG and NCES developers for decades into the future.

- Foundational research on the theory and practice of designing high assurance protocols that can be verified mechanically, a methodology that slashes the risk of human error and can also improve programmer productivity. In particular, we showed how our software can be integrated with a powerful automated theorem prover, NuPRL, which can then be used to guide the creation of highly robust executable code with provable properties and a close match to the properties of a target runtime environment.

- A new approach to building scalable systems using protocols adapted from the peer-to-peer computing arena, and several important proofs of the concept that have already transitioned into major industry cluster-computing products and platforms. In particular, Microsoft’s Windows Clustering system, in the Longhorn release, employs this technology. The key ideas here were to integrate peer to peer architectural elements with a style of communications protocol called a “gossip epidemic”. We found that such epidemics are extremely robust and found several ways to employ them: for replication of slow-changing data, for repair when inconsistencies develop among systems over time, to orchestrate self-management and self-repair, and even as a part of a scalable reliable multicast protocol.

Our work also entails assisting the military and other government agencies in defining new research programs and architecting their GIG systems. We’ve worked closely with the Air Force Joint Battlespace Infosphere (JBI) team and have developed productive collaborations with military vendors that range from IBM and Microsoft (mentioned above) to Raytheon, Lockheed Martin and Telcordia. We helped the AFRL JBI team brief the AF SAB on scalability challenges and related QoS issues in the JBI, and our ideas are prominent in the thinking for JBI’s Prometheus design and build. Moreover, DARPA created two funding programs to explore consequences of some of our work: The IPTO SRS program, under Lee Badger, builds on work done in our effort, and the IPTO SAPIENT program, under Jonathan Smith, seeks to take next steps in the area of compositional protocols, inspired in part by our accomplishments.

We’ve also maintained an active dialog and visibility within industry standards groups, such as the W3 consortium (developing Web Services standards) and the Autonomic Computing Consortium, exploring self-management and self-repair options for industry platforms.

Finally, we created a company that worked on technology transitioning activities until the 9/11/2001 terrorist attack. Unfortunately, the company was a victim of the technology sector downturn triggered by that event and closed its doors early in 2002.

The remainder of this document tracks our effort in rough chronological order. We then provide a detailed discussion of some of the major technologies we developed and tables with follow-up information for some of the transitioning events mentioned above.

CONTEXT FOR OUR EFFORT

Our project began in mid 1999, when the Internet boom was still gathering speed. At that time it was already evident that commercial trends overlooked many of the most important needs of mission-critical military and non-military computing. The Web was fast becoming the ideal environment for selling merchandise of all forms, but was inhospitable to applications designed for such purposes as monitoring a sensor network to detect suspicious activities, assisting a soldier in finding ammunition or shelter in the heat of battle, operating the nation's electric power grid, or securing the privacy of sensitive medical records. Even the security mechanisms of the Web were dangerously limited, focusing almost entirely on commercial transactions and overlooking the diverse but special requirements of military and sensitive commercial security applications.

Our goal was to use new styles of distributed computing systems to overcome some of these limitations and to show that even applications built using completely standard approaches and tools could benefit from the mechanisms we would develop. We submitted two proposals to the DARPA FTN program under Doug Maughan: one to push beyond the state of the art in the area of scalability and reliability, and the second explore the use of formal methods as a way to automate some of the most tedious and error prone aspects of building optimized communication protocols for use in demanding environments. Funding was approved, but DARPA requested that we merge the projects, the first of a series of administrative decisions that make the current final report complicated, because it must report on multiple distinct activities and a range of publications that ultimately grew to include 78 published papers (almost all in major international conferences or top-ranked journals), many keynote talks and other prestigious speaking invitations, and most recently, a book aimed at masters-level graduate students. That book would make a very reasonable final report for our work; this short summary is necessarily too brief to do real justice to the many results achieved, and must limit itself to highlights.

In the subsections that follow we review the work done by our two primary sub activities. Later in the document we provide a "fact sheet" for each technology mentioned, explaining the technology in more detail, giving references to published papers, and pointing to concrete transitioning activities. But these sub efforts weren't the entire story: after completing our basic FTN effort, we were provided with Seedling funding for a project in DARPA's new SRS program (Self-Regenerative Systems). Thus our single contract grew to include a third distinct "project".

SUBACTIVITY 1: SCALABILITY

Our first subtask explores a new way of building distributed systems that aims at taking reliability, security and integrity to a completely new scale. This effort (associated with a class of peer-to-peer technologies using epidemic protocols) ultimately yielded both a methodology for solving such problems [35] and a series of real systems, which we describe in some detail below – systems that we've named Bimodal Multicast [78], Astrolabe [31], and most recently, Kelips [22]. These systems are intended to be both useful in their own right and also interesting as examples of how the methodology can be applied to concrete problems and how the solutions can be integrated with standard software platforms.

Bimodal Multicast aims at a well known problem in reliable communication: existing reliable multicast protocols scale poorly [49, 54, 58, 59, 60, 65, 73, 77, and 78]. Our work explored the reasons for this poor scalability and found that overhead grows quadratically in existing protocols. Our new protocol [78] uses epidemic (gossip) communication in a peer-to-peer configuration and demonstrates that with this approach, it is possible to send messages reliably to very large numbers of recipients. Epidemic methods for database replications were first proposed in work by Demers and others at Xerox Parc but had not previously been used for communication protocols.

Bimodal Multicast pushes well beyond the limits of any previous multicast technology and does so in several respects: numbers of recipients that can be accommodated are substantially in excess of what could be done with prior approaches; the level of reliability achieved in very large configurations is very high and increases with system size, whereas with prior technologies reliability decreases as a function of scale; the ability to tolerate network disruptions or participant failures has been enhanced so that a larger number of faults can be survived; and the stability of data delivery is quite good even when the network is somewhat disrupted or some recipients fail during a run of the protocol (important in many media applications, such as video or voice).

These are ambitious claims and we don't make them lightly. To evaluate and demonstrate the properties of Bimodal Multicast, our project started with a detailed baseline evaluation of the best pre-existing multicast protocols (virtually synchronous multicast, Scalable Reliable Multicast (SRM) and Reliable Multicast Transport Protocol (RMTP)). We documented the scalability problems that arise in these protocols [59, 60, and 78] and then revisited the identical scenarios using Bimodal Multicast to explore precisely how our work stacked up next to the prior work. In doing this, we found it necessary to develop a comprehensive theory and analytic model for the protocol, we implemented a detailed simulation, we implemented the protocol itself in several contexts, and we conducted experiments in real and emulated network settings [59, 60, 65, 73, 77, and 78]. The actual software was also made available to others and has entered general use through the Ensemble platform distribution from Cornell, at no fee.

Every communications protocol has vulnerabilities, and while Bimodal Multicast overcomes weaknesses that cause serious problems in many existing products, it also brings some new limitations that need to be understood, and perhaps reduced through additional research. The main disadvantage of the approach is that when a machine comes under extreme stress, it may drop incoming messages (the gaps are reported but after a brief attempt to repair them, Bimodal Multicast will give up). Our work remedies this limitation by using application-level logging and a log-based recovery protocol, and we explored the benefits and costs associated with such a technology layering, notably in [55, 59].

Scalable multicast is just one of many ways that epidemic peer-to-peer protocols can be exploited. *Astrolabe*[1] solves what might be called the "inverse" problem relative to Bimodal Multicast: rather than focusing on a "few to many" multicast model, Astrolabe asks how best to deal with settings in which every computer is generating data and everyone potentially needs to access summaries of that data [28]. That is, the system offers a new way to monitor and control a large network composed either of sensors or of application programs distributed on very large numbers of computers. The approach we took is remarkably flexible: Astrolabe offers a self-configuring and self-regenerating distributed database to the user, and can be queried or even reconfigured as needs evolve, in real-time. This makes the technology a good choice for distributed problem solving, intrusion detection, knowledge discovery, data mining and fusion, or system management and control. Publications exploring these cases include [1, 6, 8, 9, 10, 11, 16, 20, 27, 28, 29, 31, 32, 33, 35 and 37]. We recommend [28] as a primary reference for those interested in learning more; the remaining papers tackle "secondary" issues.

Broadly speaking, the role of Astrolabe is to track a class of distributed information being generated in an application spread over a network, or in sensors located at multiple places in a networked setting. Traditionally, networked applications have operated in the dark: we've lacked tools to help them configure themselves, repair themselves after disruption or failure, and even to gather desired state information in a consistent, scalable manner. A result is that applications implement all sorts of ad-hoc mechanisms to detect failures and react. Surveys of these mechanisms reveal that many are relatively ineffective, and few guarantee consistency either in the way that conditions are sensed or the forms of response that an application might attempt.

[1] With respect to Astrolabe, we note that the software was actually developed by the investigators in a startup company (Reliable Network Solutions Inc), now defunct, and that none of the source code or development activity for the system per-se occurred at Cornell. No government or Cornell funding was expended on this development activity; it was undertaken outside of Cornell and funded from venture investment and the company's revenue stream. Unfortunately, Reliable Network Solutions was caught in the downdraft after the 9/11 catastrophe; the venture investors pulled their funding out days after 9/11 and the company itself failed a few weeks later when it was unable to line up new funding. The technology was passed into a holding company, called Web Sciences LLC, and a license was written to give Cornell unlimited research access to the technology. (In contrast, the other software systems mentioned in this report: Bimodal Multicast, the older Ensemble and Horus systems, Kelips, Willow, etc are all Cornell-developed, and are all available for users from Cornell, in source form, under a non-fee-bearing license.) We do not anticipate any difficulty in obtaining additional licenses for use of Astrolabe or inclusion of Astrolabe into QuickSilver, should the need arise.

Astrolabe accomplishes these tasks using a robust, scalable protocol that imposes negligible load on the participating computers (just a message or two every five seconds or so), has a very small use of memory on these nodes too (a few k-bytes), and creates the illusion of a high quality database in which system state is conveniently represented and updated as changes occur. We've also looked at security for Astrolabe. The basic system uses public keys to secure itself against many forms of attack. Nonetheless, security for some aspects of Astrolabe remains an open problem and we hope to do more work on the topic in the future. Astrolabe shares the stability of Bimodal Multicast when placed under stress.

Much as in the case of Bimodal Multicast, we evaluated Astrolabe carefully and compared it with prior "baseline" scenarios through exhaustive and detailed theoretical, analytic, simulation and experimental studies. With the AFRL JBI team we explored the validation of Astrolabe in military settings of interest and importance, and through collaboration with the AFRL/Cornell Information Assurance Institute we investigated the addition of strong security mechanisms to the base Astrolabe technology. Finally, we made the technology available under a no-fee license, and plan to make use of it in our new integrated platform, QuickSilver.

Kelips, developed under SRS seedling funding, explores yet a third positioning of the same core idea [2, 18, and 22]. Kelips is a new indexing tool for tracking information down in a very large network where many computers may have information to publish, and many users are doing queries. Kelips shares the robustness and scalability of Bimodal Multicast and Astrolabe. In technical terms, Kelips is a new "distributed hash table" (DHT), distinguished from all prior DHT's by its exceptionally fast lookup times (a lookup completes in O(1) time), its rapid adaptation when participants join and leave, and its use of probabilistically replicated data to obtain exceptional resilience to disruptive events [22]. Kelips is like Astrolabe in using very little communication to accomplish its goals – just a message or two every few seconds – and rather little memory. Like Bimodal Multicast and Astrolabe, Kelips has some weaknesses that will need additional study. In particular, we have yet to look closely at the security properties of the system. Moreover, we have identified conditions (those involving very rapid updates system wide) in which Kelips can lag the update rate and begin to report changes to the index in bursts. So far, however, neither class of problems seems to be at all common [2, 18].

Willow and Selectcast, also developed under SRS seedling funding, are overlay multicast technologies that play a role similar to IP multicast, but operate "end to end" (no router support is required) [6, 12]. Willow exploits protocols similar to those on which Astrolabe was based to adapt its overlay rapidly and accurately when receiver interests ("subscriptions") change or when failures occur. SelectCast is an earlier version of Willow in which Astrolabe itself was used for these purposes [6, 12].

In addition to developing these software platforms, we've applied these ideas in a number of realistic settings, and developed very detailed simulations as well as implementing the technologies and using them to solve real-world problems. This has resulted in a number of technology transition opportunities, which are now being explored:

1. Amazon.com, Yahoo, Google, Lockheed Martin, Raytheon and the companies that operate large sectors of the US electric power grid have all been in contact with us about use of our technology in demanding, large scale settings. For example, Amazon operates data centers with 4000 computers online today and as many as 10,000 expected within 18 months. Such settings are rich in the kinds of problems on which we've been focused and also offer an opportunity for us to see Astrolabe "in action", a kind of study that could help us understand where to take our work next. Raytheon, our partner in the new SRS program, will play a similar role with respect to challenging military applications. We are also in dialog with Lockheed Martin, MITRE and other companies about use of our technologies in other kinds of very large-scale military and sensor systems on which they are working.

2. The Air Force has used our work as the basis of its proposed architecture for the Joint Battlespace Infosphere, or JBI. We are now working closely with AFRL on developing a JBI prototype which could be used both for JBI operational experiments and also as a form of blueprint for commercial JBI products. We have been partners with the AFRL JBI team in developing the system specification and API and are particularly involved in aspects of the JBI concerned with scalable publish-subscribe data dissemination. In this connection, we note that Reliable Network Solutions, a small company created to explore military and other transitioning opportunities, was caught in the high-tech downturn and failed late in 2002. However, we are now exploring the possible creation of a follow-on company which could pick up where RNS left off, perhaps through some form of partnering relationship with Raytheon, Lockheed Martin, MITRE or a similar company.

3. Microsoft has integrated several of our technologies into parts of its Windows XP product line. For example, our work is the basis of a next generation of the Microsoft clustering product, NT Clusters, which (when it reaches the market) will offer far better scalability and data center administration tools than are now available in that system. Microsoft has also started a major development effort seeking to build a large-scale publish-subscribe technology using ideas drawn directly from our work on Astrolabe and Bimodal Multicast. Microsoft is supporting some of our students, has collaborated with us on jointly authored papers, and often cites us as a center of excellence for academic work on security and reliability.

4. IBM has launched a major initiative to develop new tools that will make computing systems much more "Autonomic" – self configuring and self-repairing. As noted earlier, IBM is using our older work on data replication in their WebServices product, and we are now in dialog with the company about similarly using Astrolabe in WebSphere. We believe that Astrolabe could offer Web Sphere developers a powerful new set of options for large-scale state monitoring and representation. Of course IBM currently owns Tivoli, hence it is not clear that Astrolabe per-se could be used for this purpose. We are in dialog with the company and have spoken at several of IBM's major Autonomic Computing workshops.

SUBACTIVITY 2: PROTOCOL CONSTRUCTION WITH COMPOSITIONAL METHODS

Recall that our effort has two-sub efforts. The subsection above reflects the outcome of our work under FTN funding, in the area of distributed systems scalability. But that was just one of two funded proposals in the FTN program, merged into our single DARPA/AFRL contract.

This second subtask emerged from an older DARPA effort, funded during the period 1995-1998. At that time, our group was known primarily for its work on security in groups of cooperating programs and for work on data replication and fault-tolerance in such group settings. We developed a series of systems for this area: the Isis Toolkit (1987), the Horus system (1995) and the Ensemble system (1999). The work became the basis of the fault-tolerance standard used by the CORBA architecture and has been widely used, copied, and commercialized. For example, the Isis Toolkit itself is still used at the core of the New York and Swiss Stock Exchanges, the Naval AEGIS battle control and communications system, and for several purposes in the French air traffic control system. The developers of the Boeing 777 SafeBus system have often described the system as being based on the Horus architecture. Moreover, the virtual synchrony model employed for data replication in these systems is now widely viewed as a standard for high-speed data replication.

Very recently IBM adopted Cornell's virtual synchrony architecture and platform as the basis for a new "Distribution & Consistency Service" in their WebSphere product, currently the market leader in the "Service Oriented Architecture" (SOA) area. WebSphere is widely seen as a likely majority platform for DoD GIG efforts. WebSphere version 6.0 will permit the development of Web Services offering flexible levels of availability, security, and other properties using a compositional, layered, architecture in which the properties of a service can be closely matched to requirements. This will bring a DARPA-developed technology into the hands of very large numbers of developers and could greatly enhance platform quality of service guarantees for DoD and other applications in years to come.

However, the success of virtual synchrony as a model and of these specific systems as programming tools also forces attention to the correctness of the tools themselves. For example, if group replication is used to manage a replicated security key, a security flaw in Ensemble might compromise all users of that key. A bug in our protocols could potentially block the reporting of updates, leaving applications with stale state information or otherwise hung. Thus, replication mechanisms can be a two-edged sword, creating serious system-wide security and reliability exposures.

Our project tackled this question by using formal theorem proving tools to develop a methodology for greatly increasing the level of assurance associated with critical properties of systems like Ensemble. The hope was to formally specify the key properties of our system, then use theorem proving tools to analyze the code and demonstrate that it achieves those properties and that it would tolerate the classes of failure that arise in real-world scenarios.

This effort benefited enormously from collaboration with two other groups. One, at MIT, had developed a new way of describing the properties we might wish to verify (the so-called I/O Automata approach, pioneered by Nancy Lynch under DARPA funding). The second, headed by colleagues of ours here at Cornell, had developed a theorem proving tool that can directly manipulate code in certain programming languages as well as specifications coded using Lynch's IOA formalism.

Our approach started with the Ensemble protocols used to replicate data [73, 76]. In Ensemble, these are relatively simple protocols, but they are then subjected to a series of semi-automatic optimizations before actual machine-code is generated, and the output of this process is relatively complex. Human inspection of such code is difficult. Accordingly, we set out to develop proofs concerned with the correctness of the optimizations. These are code transformations used primarily to ensure that the performance of the system will be competitive with the best hand-optimized virtual synchrony implementations. At the same time, we hoped to demonstrate that our code correctly implements the virtual synchrony specification with respect to the replication of security keys.

When we began this work, we recognized that if we simply tried to develop the desired proof without any simplifying methodology, the undertaking would be on a scale dwarfing anything previously attempted by the program verification community. Accordingly, our work occurred in two stages. First, we developed a new compositional proof methodology whereby proofs of smaller components can be combined to prove properties of larger systems. In the case of Ensemble, this was particularly feasible because the protocols we focus on are very regular in structure, are small, and are assembled into stacks by a form of protocol composition. Then, we actually built compositional proofs of the desired Ensemble properties.

Our effort was quite successful [48, 53, 57, 69, 73, and 76]. Ensemble is today the most strongly assured data replication and group communication technology available. Our work identified some very subtle bugs, assisted us in developing some complex code optimizations which might otherwise have greatly complicated the protocols themselves, and yielded a verification methodology that has rapidly captured the attention of the formal tools community. The resulting system is extremely robust: users do report bugs associated with the compilers used on some platforms, and have asked us to look at problems stemming from oddities of one operating system or another. Yet over the five years Ensemble has been in active use, we've had essentially no reports of bugs in the replication or security mechanisms of the system, at all. This doesn't rule out discovering such problems later, but is certainly reason for cautious optimism.

As in the case of the first half of our effort, we have published extensively on the Ensemble verification work and made the software available to those interested in using it (in source form, free of any restrictive language or fees). Ensemble itself has been transitioned into many settings.

1. At DARPA, Jonathan Smith's SAPIENT program reflects some of the ideas and successes of our compositional networking approach. SAPIENT will apply similar

approaches (composition of microprotocols with strong formally specified semantics) in settings such as communications networks targeted to military systems in the field, where it is important to match the protocol to the property of the communications environment.

2. IBM has run into fault-tolerance and fail-over challenges in its WebSphere product line and is developing a new high availability architecture for WebSphere using our Ensemble system as its basis. Moreover, Ohad Rodeh, within the company, is providing support for Ensemble users worldwide.

3. Left Hand Networks, a telecommunications and storage-area networking company has developed a commercial version of Ensemble and is using it as the basis of a technology for controlling large numbers of attached storage devices using Storage Area Network (SAN) architectures. IP-SAN, this new technology, is unusual in supporting very flexible administration of the storage device pool, and in adapting more rapidly than any other technology when devices are taken offline (or fail), or brought online. Moreover, the security features of the technology are employed in protecting devices against intrusion. Mark Hayden, one of the original developers of Ensemble, heads this activity at Left Hand Networks. The product line is called "Distributed Storage Matrix" (see www.lefthandnetworks.com).

4. Nortel, the Canadian-based telecommunications giant, is building new technologies for large-scale collaboration and switch control. Early in 2000, the company copied the entire Ensemble and NuPRL code base and since that time has been extending and using it internally. Nortel has treated the effort as proprietary and we are not sure what its current status is.

5. There are a large number of academic researchers who use Ensemble for teaching and research applications. Several formal verification systems are now using the kind of compositional component-oriented verification we first demonstrated in our work on Ensemble. Moreover, Ensemble itself is used by NuPRL as a mechanism for sharing the computational burden of verifying a large proof over a pool of participating machines.

WIRELESS COMMUNICATION TOPICS

In this subsection we report on some work undertaken as a “spin-off” from our basic effort. Although not part of what we originally proposed to DARPA, the widespread adoption of wireless devices (notably 802.11b cards) forced us to look at issues that might arise in using Spinglass technologies on wireless LANs. This work resulted in several papers:

1. New routing algorithms for ad-hoc networks in which the same sorts of peer-to-peer epidemic (gossip) protocols employed by Astrolabe and Bimodal Multicast are used to enhance the quality of routing [58].

2. A new “adaptive transport protocol” supporting priorities and deadlines for communication over a wireless link with variable connectivity [23].

3. A new mobile file system designed to take advantage of our adaptive transport protocol to perform caching and prefetching in ways that reflect awareness of variable connectivity to the wired network [11].

4. A new “self-centered” approach to tracking the status of nearby sensors and resources in mobile applications. For example, a soldier could track the availability of ammunition and supplies in his vicinity, or a firefighter responding to a forest fire could track the humidity and temperature conditions in the region around him. (This is basically a new implementation of Astrolabe for wireless settings) [1].

5. A radical new approach to “time sharing” wireless 802.11g communications cards in settings where there may be multiple networks, for example because of security or infrastructure/ad-hoc considerations. Microsoft is incorporating these ideas into its wireless software architecture; hence they will reach the consumer at no additional cost beyond the cost of the basic Microsoft platform [4].

Relatively little DARPA funding was expended on these mobility and wireless topics, in part because we looked at relatively narrow questions, and in part because the topic attracted industry support. Microsoft, for example, provided support to two of the graduate students who looked at these questions. In effect, we were able to use our FTN funding as leverage to tackle what turned out to be important questions on the periphery of our primary topic.

SRS SEEDLING RESEARCH

Although readers will surely find this confusing, our project actually had yet *another* significant component. Up to now, we've discussed work done under the DARPA FTN funding that began in 1999, pointing to two major sub activities and some additional but relatively minor work that spun out of these core efforts. However, early in 2003, our FTN work wound down (in fact, we submitted an early version of this report at that time, prior to the close down of Doug Maughan's effort). These varied threads were unified for administrative convenience, and as a result we are compelled to report on all of them in the present report, even though they would more naturally be treated in completely distinct reports.

Our SRS Seedling project focused on the right way to bring the accomplishments of our prior work to bear on the real problems confronting today's military branches. We looked closely at the JBI and asked whether we could use our technologies to break through the scaling barriers the JBI confronts. Then we stepped back and asked whether the resulting system could be understood as a form of platform over which an application runs: the JBI as an application, in effect. QuickSilver is our platform, and when it is completed we believe it will host many such applications.

The seedling effort had several kinds of outcomes. First, we identified some technology gaps [8]. For example, to build the JBI's "data repository", we needed a distributed indexing technology, yet existing indexing schemes scaled poorly and were easily disrupted by stress. This kind of thinking motivated the development of Kelips, mentioned above: the world's most scalable and fastest distributed indexing system. Concerned by reports that other peer-to-peer research groups were having problems with overhead triggered by churn, we developed a "churn test" for Kelips, and found that with minor changes to the protocol, it could be made almost completely immune to these kinds of problems.

Similarly, we discovered a need for an overlay multicast communications layer for QuickSilver, and asked ourselves how best to build such a layer. After experimenting with a solution based on Astrolabe, we concluded that while Astrolabe is wonderful for many purposes, this was not one of them; the technology wasn't reactive enough and our multicast data streams were too easily disrupted by failures. We therefore developed Willow, a new implementation of the ideas seen in Astrolabe, but rearchitected to perform much better under the sort of failure scenarios that proved problematic when building overlay networks using Astrolabe to sense the system state. And this was a great success. Willow will be an important component of QuickSilver.

Our seedling work also led to a recognition that we needed a better testing and evaluation platform. For many years, we've tested our software using the Cornell network as an experimental setting, but we have little control over the network as a whole or the loads on it. We found that it was too hard to set up experiments and too manually-intensive a task to collect the results, inject failures or stresses, mimic specific network setups of special interest, etc. Such thinking resulted in a DURIP proposal under which we obtained a 256 node cluster capable of emulating very large networks. With SRS bridge funding, we ported emulab software to the cluster and also developed our own runtime environment for experiments stressing our protocols in varied ways. This allows us to undertake detailed before and after comparisons, to measure the performance of our software with great precision, and hence to demonstrate that the work we are doing is having the desired outcome (or, if not, to hone in on the source of problems).

CONCLUSION: A TECHNICAL VISION FOR GIG/NCES PLATFORMS

Since the single AFRL contract was actually used as a vehicle for two separate research activities in the early FTN effort, and then became the vehicle for SRS Seedling funding, this compels a certain form of presentation in which each of the things we accomplished was "credited" against the corresponding funding source.

But in fact our work has a greater degree of coherency than is suggested by such a review, and the goal of this subsection is to weave a "unified technical vision" out of the varied parts. In doing so, we'll back up to the early period and focus not so much on the individual systems we built but rather on the technology context that led us to tackle each problem at the time and in the order mentioned. In effect, we want to ask: what were the pressing military and scientific knowledge gaps that motivated our efforts? In what ways did our work respond to these questions? How can we demonstrate impact on the military and on the available technology options on the shelf today?

To answer such questions, we'll start by restating the overarching theme of our project through this entire period and over the various sub-efforts we've mentioned: The Spinglass project was created to explore a new generation of reliable, secure and scalable technologies for distributed computing systems. Clearly this is an important goal: the DoD GIG and NCES visions demand technologies having these properties and the government is already deeply committed to transforming American forces in ways that presume the success of the GIG undertaking. Yet commercial off the shelf products lack all of the needed properties!

The broad problem we're facing, and that the Cornell effort is trying to solve, is to break through the technology barriers that, if left standing, could prevent DoD from achieving its goals – the country's needs – in the GIG/NCES arena. Consider the characteristics that future GIG systems must exhibit:

- Support large numbers of users – they need to "scale" well. Scalability actually has many dimensions: a system that works well in the lab with ten users should work well in the field with 10,000. The system should work as the size of the network grows, exposing it to increasing rates of network problems and higher latencies. It should scale well in the loads on the application: if we want to double the capacity we should be able to do so by doubling the amount of hardware. And the administrative costs of running the system should grow very slowly as we scale the system up.

- We need ways to avoid dependency upon centralized servers or other single points of failure. This is especially true in military systems, where centralized "single point failures" stand out as high-value targets.

- We need ways to offer various quality of service properties, and the utmost in security. This is important because one doesn't see the same mixture of needs in commercial settings; hence the commercial vendors aren't paying enough attention to such issues.

- These systems provide value by offering new ways of sharing information or other forms of coordination through the sharing of a consistent view of distributed system state. Thus, consistency is an important issue, and we want to ask how inconsistency impacts behavior. Could an intruder disrupt a system by compromising just a node or two? How expensive is it to protect against inconsistency and to repair problems when they occur?

- These systems will need sophisticated information architectures, extending to properties such as trust and security as well as to relationships between types of information. We will need to know that users can trust information, can trust that others have been correctly authenticated, must respect any need-to-know access policies imposed by a system commander, and must not themselves become a vehicle for attacking the platforms connected to them.

- Must offer robustness against the forms of disruption, failure, and attack that often arise in demanding settings such as battlefield scenarios. Disruptions could come from crashes but also damage, and in some cases, insider attacks.

The basic premise of our entire project is that while the GIG and NCES vision of communication isn't really such a new development, the mixture of special requirements just enumerated takes us far outside of the commercial product space. *Unless we can solve these technical problems and transition the solutions into COTS products, important classes of GIG and NCES systems will fail.*

To give a concrete example: the GIG architectures make considerable use of publish-subscribe technologies, and there are many publish-subscribe products on the market from sources such as IBM, TIBCO, Vitria, and Java Soft. Yet none of them has the mixture of scalability, robustness and security just cited. *None even comes close.* These forms of weak products and solutions thus create a serious problem – they invite the application developer to embark on a kind of architectural design that may succeed in a superficial sense and yet yield a deeply flawed solution that attackers can easily disrupt or disable.

The use of distributed computing technologies in demanding military, government and commercial settings involves overcoming a whole series of technical obstacles not encountered in smaller-scale networks. Here, when we talk about a small network, we have in mind a system with perhaps 50 or 100 computers, tightly coupled on a communications device like a shared ethernet or a wireless network. Such systems have very predictable bandwidth, latency and throughput properties and are often secured by firewalls or VPN technology. The problem changes dramatically if we want to treat a very large collection of computers as part of a "single" computing system. Now, we face the challenge of coordinating the behavior of perhaps hundreds of thousands of computers. Even if the number of machines involved in a particular interaction is much smaller, the secure, efficient control of a massive network takes us into a domain that distributed computing has heretofore overlooked.

Viewed this way, emerging military systems will be extremely large. All four services are migrating to secured IP infrastructures, which makes it feasible to design applications that might be used by thousands of combatants, or that gather and synthesize data from thousands of sources. The benefits of such a step are clear; the challenge is to end up with solutions that really work. As is evident to anyone who uses the Web to locate news articles, the most standard ways of building network applications are completely inadequate for these kinds of mission-critical applications.

Examples of the kinds of technical obstacles we face are:

- Scalability. Not many of the existing technologies scale particularly well, especially if scalability is construed broadly to include steady performance, predictable and low jitter (variance of throughput), fault-tolerance, etc. We need to know that if a system works well in the laboratory with a few dozen users, it will also work well in the field with hundreds. Thus we need to create a new science: a science of scalable protocols.

- Security. The larger a system, the greater the potential for attacking an enterprise through that system. Anything one would imagine installing on hundreds of thousands of computers needs to be secure to a degree rarely encountered in prior work. Not only must our protocols be scalable, we need to show that their properties can be maintained under attack and that they can protect sensitive information.

- Firewalls. A very large system will need to tunnel through various kinds of firewalls. While firewall tunnels are a familiar technology, they can only be used with great care.

- Flexibility. The needs of applications and styles of use of our large system will change over time; the technology needs to offer a growth path to the user.

- Ease of management. A large-scale environment is in constant flux; only a technology offering a high degree of intrinsic robustness to fluctuations in the behavior of the underlying network will prove stable enough to operate correctly without constant human intervention.

- Fault-tolerance. In a large-scale setting, we face high aggregated rates of crashes, restarts, network reconfigurations, and other disruptive events. The technology needs to offer reasons that would lead us to feel confident in its ability to ride out such disruptive but transient failures.

To reiterate the point just made, existing distributed computing products lack so many of these properties that without progress we will simply be unable to build and deliver the GIG and

NCES solutions we need. While there have been isolated success stories, they typically involve very specialized systems used in very cautious ways. The scenarios just surveyed would break any existing technology. Needed are radically new approaches to very large scale networking.

Our project is part of a DARPA-initiated response to the need. Spinglass, and now QuickSilver, were conceived as efforts to show how a new approach to networking, based on "gossip" communication protocols, can break through the limitations that have stymied so many commercial product offerings. Spinglass tackled the issues one step at a time, carving off specific technical challenges, solving them, demonstrating the solutions in the context of real software systems that people can actually use, and then transitioning the solutions into more integrated off-the-shelf technologies by partnering with companies like IBM, Microsoft, Raytheon, Lockheed and others to ensure that when we come up with something that the GIG and NCES development communities will need, they can find the solutions in familiar, high-productivity forms and standard products.

Viewed this way, we can revisit our work chronologically, and see the various parts of the effort against a single longer term goal.

To illustrate these ideas, consider a military tactical information system such as might be used in a coalition engagement in a setting like the Balkans. The network hosts a great number of computers, which could be loosely partitioned into sensors that "detect" things, command and control systems that gather large amounts of information, synthesize the results, and share them with users, and end-user systems that query the environment for information.

For example, a patrol might request updated intelligence information about a town ahead, behind a hill. Abstractly, this information could come from many sources – many sensors. Perhaps there are high-flying drones or other aircraft with radar and other signal intelligence information. A satellite may be in position to image the area. Intelligence analysts may have updates to maps, showing bridges that have been taken out, the locations of defensive installations, and so forth.

Using our technology, Astrolabe might be employed to report the locations for which the various sensors have information. Multicast could be used to transfer maps and other information from the intelligence database to the computers of mission commanders. Later, as the mission progresses, the computers used by the participating soldiers could use Astrolabe to report status changes – location, health status of the personnel involved, ammunition availability, and so forth. In the event of an injury, multicast could be used to rapidly locate a medic.

QuickSilver, the platform we're developing under SRS funding, should bring these components together in a single solution that fits industry standards and solves the military GIG/NCES need in a way that also addresses the many properties enumerated earlier.

What about the second side of our project? How can that be fit into this single unified vision?

Recall that our second line of activity is concerned with automated verification of high reliability protocols using tools such as the NuPrl theorem prover, developed by colleagues of ours here at Cornell. The idea underlying this work is that one can only have limited confidence in a system proved correct on paper, and then implemented by a team of graduate students and researchers. Even if our protocols are theoretically superior to other options, perhaps the code will be buggy!

To address this concern, we are working to show that even complex protocols can be simplified into stacks composed from simple components, that these components can be specified formally and proved correct, and that we can use theorem proving tools to automate their manipulation and to formally document their properties.

At the core of our approach is a new kind of "modular proof" which was feasible because the protocols we focus on are very regular in structure, are small, and are assembled into stacks by a form of protocol composition.

To summarize, although our effort does span a wide range of technologies and problems, these are unified by a single vision: the goal of offering GIG and NCES platforms that break through the barriers and limits associated with today's off-the-shelf solutions. The limits we're focused upon reflect deep technical challenges and making advances requires serious science. But when we've managed to demonstrate a major advance, industry has been eager to adopt our solutions. True, this isn't a style of problem that lends itself to final project reports – especially when the project is glued together from so many related but distinct "threads". But it *is* a problem of real urgency for the military, and it is a problem we can and must solve today.

SUMMARY OF SPECIFIC ACCOMPLISHMENTS

This section lists our major accomplishments, providing detail on our work in each of these areas of research and on the transitioning activities that we think are most important.

I. HELPING THE AIR FORCE AND NAVY DEVELOP NETWORK-CENTRIC COMBAT TOOLS.

Context: The Joint Battlespace Infosphere (JBI for short) is the primary “network centric” systems project underway at AFRL. The JBI is creating an Information Management System that will make all of the Air Force’s information assets – databases, sensor systems, imaging systems, mapping facilities, battlefield intelligence applications, etc., - unified within a single environment.

This said, the JBI is not so much a single system as an architecture. This architecture standardizes interfaces and makes it clear to a developer how one might build a new JBI-compliant platform, in the hope that someday there will be multiple JBI products on the shelf, competing by offering different properties or specializations, but compatible with one-another and with a wide variety of applications. At the present time there are already several JBI prototypes, and more are in the pipeline, focused on different aspects of the overall vision. There may never be a single JBI system that solves all the needs of the Air Force in a single platform.

Broadly, the idea behind the JBI is to enable a new generation of applications that can be assembled on the fly as easily as one builds a web page, drawing information from relevant servers and updating it continuously as the situation changes. In contrast, today most new applications can only be developed with a great deal of involvement by the companies that built the original information assets; thus, if such a company builds a biothreat sensing technology, it will also be assured of many decades of work interfacing that system to each application that ever uses it.

The “Joint” aspect of the JBI arises both from the idea of having many information sources in a single setting, and also from the Air Force dialog with other services. The Air Force works closely with the Navy and is hoping that the JBI might ultimately have a Navy-oriented component useful for Naval “Network Centric Warfare” applications. JBI developers are also in contact with the Army and with parts of the intelligence community. The Air Force plans to engage the Army team developing their Future Combat System and to explore commonality between FCS and the JBI. Thus, progress on this platform could have broad impact throughout the military.

For example, suppose that the Air Force needs to undertake a search and rescue mission in Afghanistan. Traditionally, either there were information tools already available for this purpose, or the necessary work would need to be done by hand. With the JBI, it will be possible to develop, on the fly, an application tracking intelligence about the stranded friendly forces, locations of enemy forces, weather conditions, SAM sites and other threats, etc. The hope is to cut the development time for these kinds of information-based applications from years to weeks and also to standardize the style of development so that productivity-enhancement tools can be created. Such thinking seeks to leverage the kinds of approaches that have been successful in the civilian sector into new kinds of military capabilities. On the other hand, in doing so, the JBI must tackle several issues (such as security and reliability) not encountered in most non-military information settings. Moreover, the large scale of the JBI (in terms of numbers of machines that would need to be connected to it for it to be useful) takes the system into a domain not yet encountered by civilian-sector technologies.

Our opportunity and role: Spinglass technology has been identified as a leading option for actually building the scalable communications and networking layers of the JBI. Whereas existing communication technologies have been found to scale poorly and to suffer from unacceptable fragility when placed under stress, lacking the desired fault-tolerance, security, and dynamic self-management capabilities for these sorts of demanding applications, the Spinglass technology suite (primarily, Astrolabe and Bimodal Multicast) was evaluated and found to scale extremely well while exhibiting the necessary properties.

In the eyes of the JBI development, Spinglass represents a major breakthrough. Our DARPA-funded effort has opened the door to actually building the JBI – an effort seen as a priority by both the Air Force and the Navy. We view this as an extremely important accomplishment, and we believe that once the JBI is operational, the technology may rank with DARPA's most visible and direct impacts on the military during this period.

Accordingly, our team has spent a great deal of time with the JBI team, working to understand their needs and to show how we can bridge the gap between our Spinglass technology prototypes and tools and the specific needs for those technologies and tools in the JBI. We've met with the JBI developers as often as several days per month, with longer workshops during the summer and the fall. Our software is in use by JBI developers at AFRL/IF in Rome, NY. Cornell also co-authored the JBI "Common API", which has now been adopted as the basis for the JBI Mercury program, an effort that will build three JBI systems and ultimately merge them into a single platform.

Impact: Whether or not Spinglass technology is used directly in the JBI (at present direct use of the technology is fairly likely, but whether this materializes depends on funding decisions that are still uncertain), our involvement has helped the Air Force break through a scalability barrier that was perceived to be threatening the entire project. This research effort continues to push the envelope by identifying research problems with a long time horizon (most of our work looks many years into the future), and it is exciting to us to have a concrete opportunity to transition the technologies we understand better into potentially important military settings.

Our overall approach thus has had a near-term and a long-term focus. Near-term, we are helping the JBI team develop a system that will work not just in the laboratory on a small number of machines, but also in the field with thousands or hundreds of thousands, with operational security requirements, and with the stress associated with an active battle environment. In our own laboratory, we are trying to understand where to go next, posing options (for example, concerning the right ways to apply scalable probabilistic protocols to support large-scale mobile systems using wireless connections), so that when we reach the stage of needing practical solutions to problems of these sorts, we'll also have insight into the best ways to solve them.

Specific Accomplishments: Up to the present, our concrete accomplishments in this area center on a series of proofs-of-concepts, including a white paper design developed at the request of the JBI team for a JBI-based Naval Sensor Networking Architecture (J-NSNA). The architecture we proposed uses JBI architectural ideas, supported by the Spinglass technologies we call Astrolabe and Multicast, to link a set of undersea sensors having very limited communications bandwidth but substantial local storage and computing capacity. Our architecture would let the Navy deploy these kinds of sensors in monitoring groups of four to eight sensors per group and allows queries to be sent to the sensors for remote analysis and data fusion, so that the limited communication bandwidth is used to maximum effect. The Navy and JBI see sufficient interest here to take the next step.

By building the JBI over Spinglass, we should be able to offer a degree of reliability not commonly achieved in these kinds of settings. Spinglass can easily adapt to route around poor communications links or outright computer or link failure, and within seconds will reconfigure itself to launch a new computation when users wish to do so. The system is also extremely scalable, so that if the Air Force or Navy needs to deploy hundreds of sensors they can have confidence that the system will continue to work as well as it did with ten sensors in a test configuration.

At the current time, we are building a prototype of the JBI publish-subscribe functionality using Astrolabe and Bimodal Multicast. This new system will be called QuickSilver, and we are hoping that it might emerge as a candidate to be one of the JBI experimental platforms.

II. BIMODAL MULTICAST AND GRAVITATIONAL GOSSIP.

Context: Existing one-to-many (or many-to-many) reliable multicast protocols lack the scalability needed to send information and urgent messages to potentially large receiver groups. As a result, it is extremely difficult to build applications in which large numbers of computers are able to track the state of critical information, such as assets and threats on a battlefield. Moreover, existing multicast solutions are easily attacked – most solutions have well-known security and reliability exposures that attackers or intruders could easily exploit to degrade the capability just as it is needed most urgently.

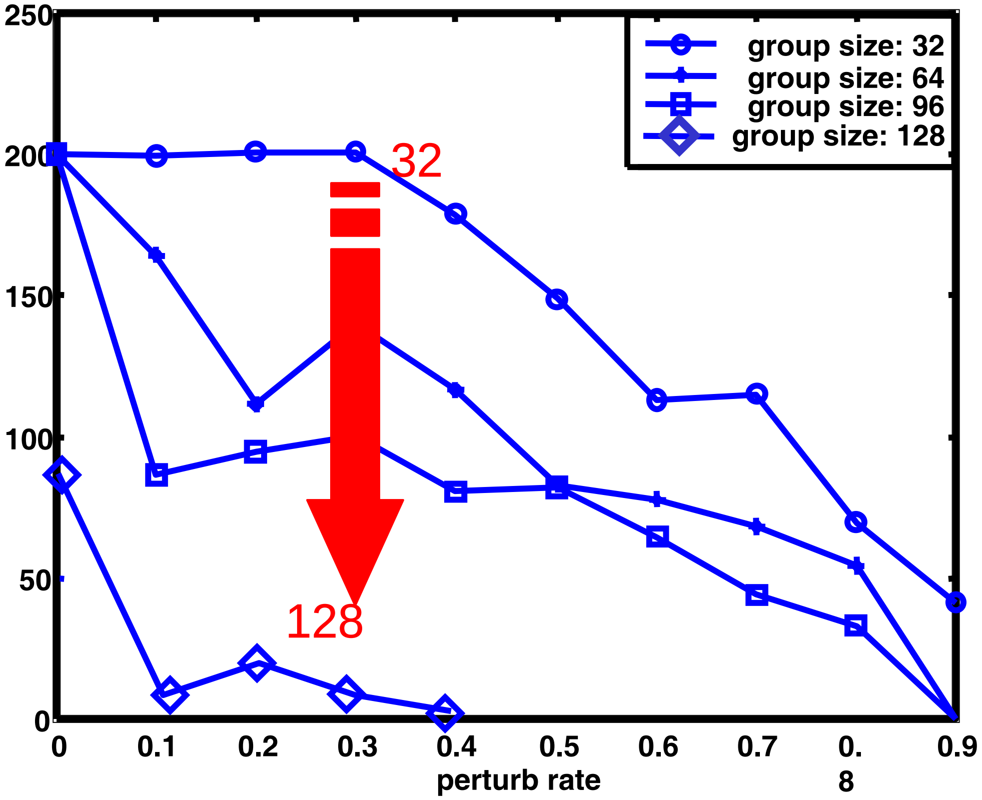

Figure 1: Multicast Group Subjected to Stress

Figure 1: Performance drops when a multicast group is subjected to stress. The larger the group the more extreme the problem. The y axis shows the sustainable data rate when 10KB messages are sent to groups of various size and the x axis shows the degree to which one of the receivers has been perturbed by stealing cycles on its machine. The other n-1 receivers are running on idle high-speed PC's.

For example, in Figure 1, we see the results of a "baseline" experiment in which we took a reliable multicast protocol (the same virtually synchronous protocol IBM is using in the Web

Sphere 2.0 system mentioned earlier) and subjected one of the receivers to a form of stress by stealing some percentage of the compute cycles from its machine. Then we graph the maximum data rate (in this case, rather large 10KB messages, with the rate measured in messages per second that can be sent without data backing up at the sender). As seen in the illustration, the larger the group the more dramatic the impact of this intervention, and indeed, with a large enough group the sustainable throughput has slowed to a crawl.

Impact: By developing new solutions and demonstrating that they can scale while maintaining a very high degree of reliability and security, commercial vendors will be enabled both to duplicate these ideas within their own products and also to open their minds to exploiting these kinds of capabilities in systems that currently lack such dissemination capabilities. As we move away from centralized client-server architectures we will gain robustness, better reactiveness, and the ability to coordinate large forces more effectively. At the same time, non-military applications in finance, disaster response, news dissemination and other uses will be enabled, bringing commercial investment into the field and enlarging the product offerings available to the military and government.

Our Opportunity and Role: The types of peer-to-peer gossip protocols we worked on in the Spinglass project turn out to be well-matched to the need. By using a fast and scalable but unreliable multicast to get the data out there and then employing peer-to-peer gossip to detect lost messages and repair the gaps, we can achieve extremely high levels of reliability in a completely scalable manner. Bimodal Multicast, which implements this approach, can also be secured against disruption, ensuring both data integrity and (because of the very large number of possible paths by which data might reach a destination) robustness when the system comes under stress.

The basic idea is easily explained, although readers seeking details will need to refer to [78] as a primary source and to [49, 54, 58, 59, 60, 65, 73, 77] for additional experiments, details on optimizations, and other findings. Basically, the protocol operates by sending messages using an unreliable multicast. Some receivers will drop such messages, since the unreliable mechanism (IP multicast) makes no effort to detect and repair problems. But each participant also tracks the membership and knows about a set of “peers”, assigned to it using a method detailed in [78]. Periodically, a participant picks a peer at random and they compare message buffers; each sends the other any messages it may be missing. In this manner, gaps are repaired and the protocol achieves a probabilistic convergence towards consistency.

For example, suppose that process P sends messages M_0 through M_k and process Q sends M_{k+1}. Now imagine that process R has missed message M_3. During each round of gossip, R will send a gossip message to some randomly selected peer and will probably also receive an incoming gossip. Thus there are two opportunities for R to discover that it is missing M_3 and to recover the data. Round by round, the odds that R will still be missing the message drop exponentially fast. Of course, we do a great deal to optimize this basic protocol, but the key ideas are already seen in this simple description.

Specific Accomplishments: We implemented the Bimodal Multicast protocol and studied it using experimental, emulation and simulation tools [78]. Since that time, we've used Bimodal Multicast as a building block in many other systems and applications. We also showed that in settings where information flows within some form of channels and applications have varying degrees of interest in the channels, a form of selective multicast can be supported (we call it *gravitational gossip* because the protocol emulates the movement of particles in a gravitational web). This protocol is particularly well suited for use in controlling the electric power grid.

Figure 2 illustrates the benefits; here we repeat the experiment from Figure 1, and with the new protocol we see that receiver data rates at healthy receivers are unchanged even as we attack the same process that caused so much trouble in Figure 1. Bimodal Multicast scales far better than any previous protocol and maintains a kind of real-time data delivery guarantee, making it especially well-suited to settings with weak real-time data requirements. Such settings are common in military applications.

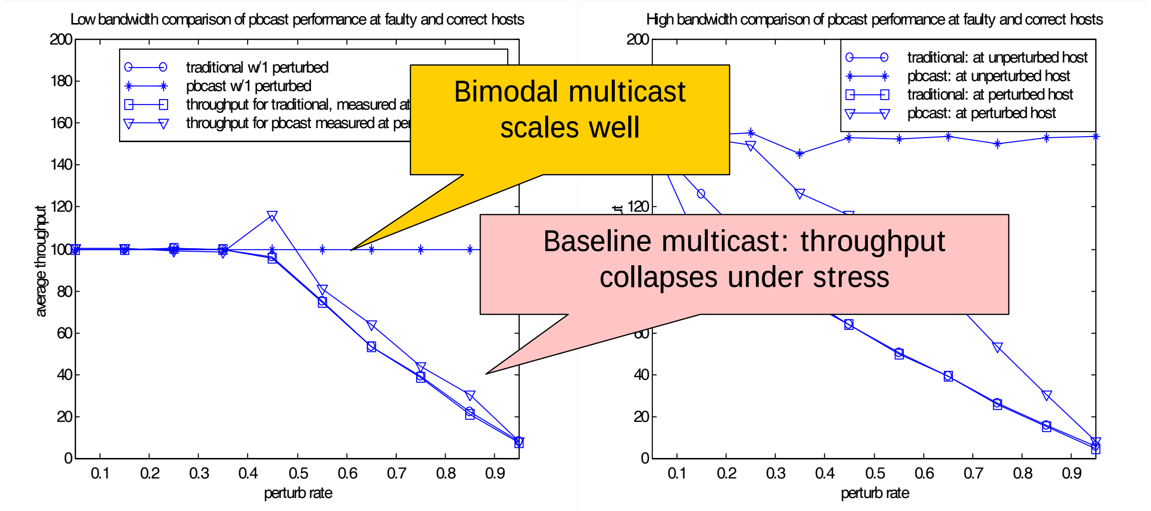

Figure 2: Bimodal Multicast performs well in the baseline scenario. In our papers, we report on other studies in which we compared with other protocols, developed formal analysis tools, undertook simulations and experimental studies, and explored other kinds of stresses on the protocol.

Figure 2: Bimodal Multicast Scaling

III. ASTROLABE.

Context: For decades, distributed applications have been designed to operate in the dark. No representation exists for the state of the network or the application, and no help is available for constructing such a state. We lack ways to gather data on a large scale, from systems which may have rapidly changing information, to data mine within such datasets, to monitor for and detect conditions of special interest or concern. We lack tools for building a picture of a network under some sudden stress, such as a disruption or an attack, and for reporting that information to the many application programs using it so that they can respond in a coordinated manner. Even problems as simple as picking the best server from which to fetch desired information can be extremely difficult because modern networks lack the mechanisms for finding out which servers are available, what data they have on them, how loaded they happen to be, etc. The main reason for this problem is that traditional distributed system architectures use a "client-server" structure, yet collecting system state at a central server can be problematic: the server becomes a single point of failure and faces a load that scales linearly with the number of sensors (or perhaps even with the amount of data they collect). This bottleneck soon becomes insurmountable.

For example, consider Figure 3, which illustrates the structure of a typical data center using a Web Services approach. A set of front-end machines dispatch incoming web queries to sets of back-end services that processes the requests in a load-balanced manner. New services can easily be added to the system at the back, and if a server gets loaded, one can just expand the cluster on which it is running.

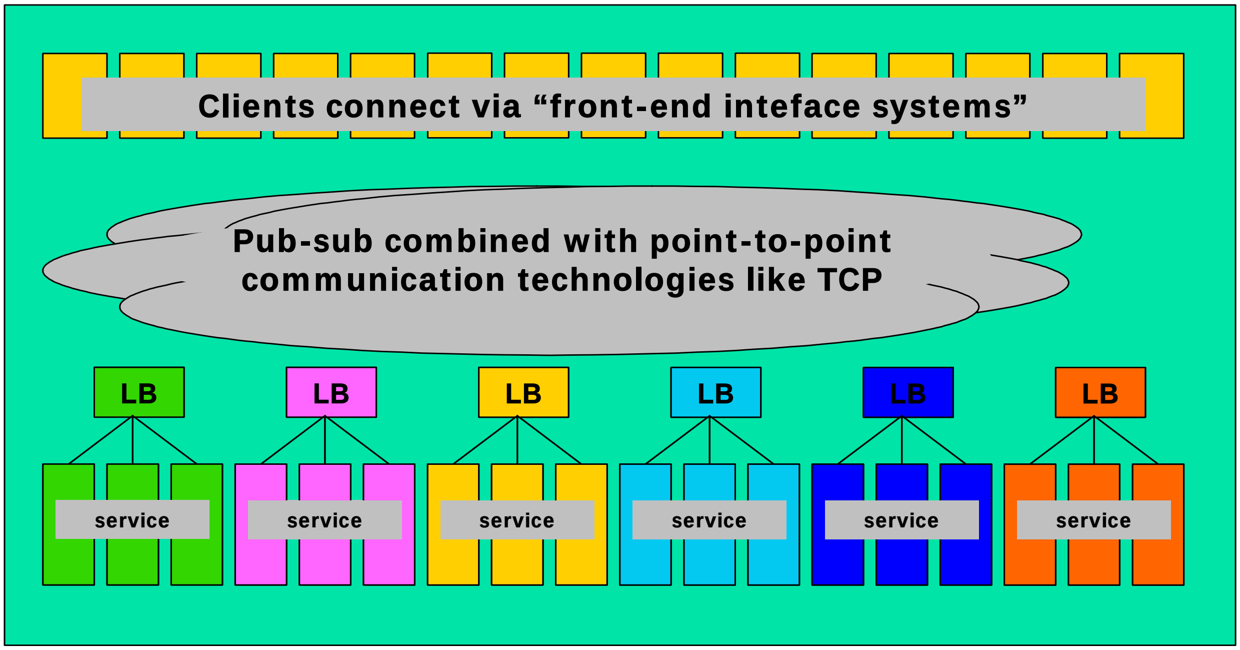
Clients connect via "front-end inteface systems"
Pub-sub combined with point-to-point communication technologies like TCP
LB
LB
LB
LB
LB
LB
service
service
service
service
service
service

Figure 3: A typical services-oriented data center

Figure 4 illustrates the same center from a different perspective: now we focus on the services themselves. We see two data centers with several services in each, each service partitioned according to some sort of key, and each partition running on a cluster.

Services are hosted at data centers but accessible system-wide

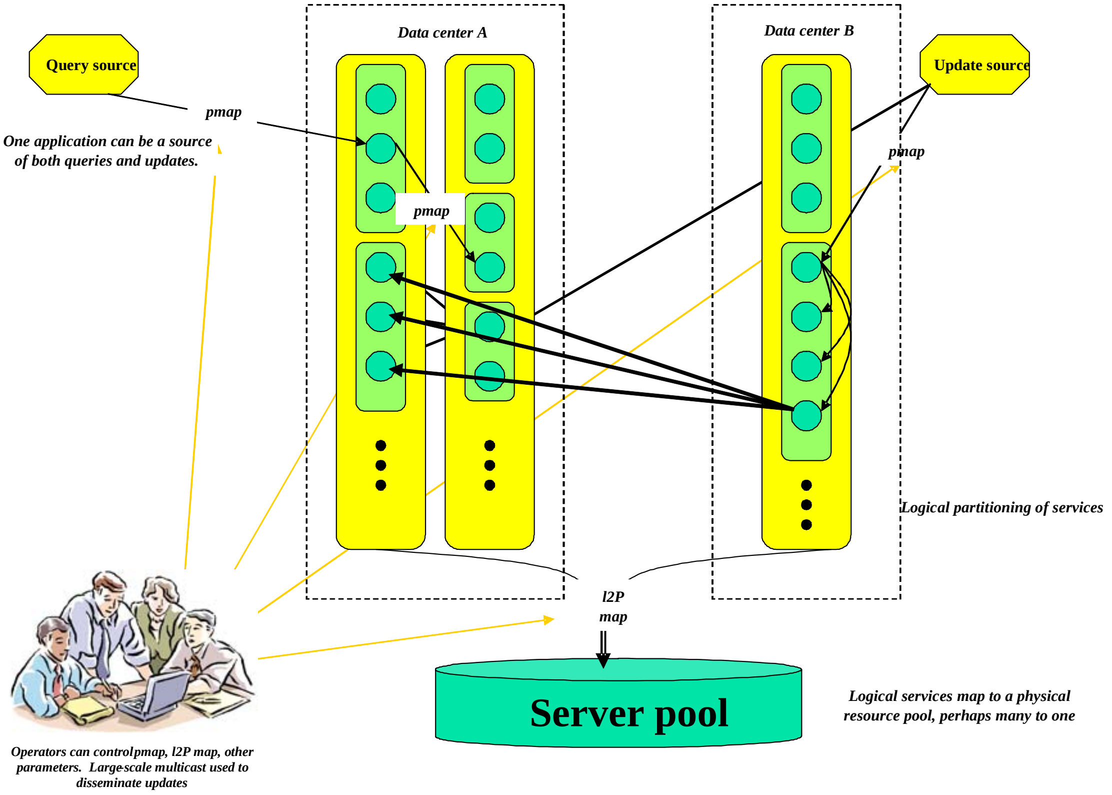
Query source
pmap
One application can be a source of both queries and updates.
Data center A
pmap
Data center B
Update source
pmap
Logical partitioning of services
l2P map
Server pool
Logical services map to a physical resource pool, perhaps many to one
Operators can controlpmap, l2P map, other parameters. Large-scale multicast used to disseminate updates

Figure 4: System-wide Access to Data Centers

Now consider just how little support is available for the developer of such services. They lack tools to automate the management of these kinds of systems, and to manage such tasks as tracking system state, load-balancing, replicating data in the clusters for high performance and fault-tolerance, tracking down problems when they occur, and ensuring that responses to queries will occur in a timely manner.

Astrolabe, combined with Bimodal Multicast, solves this problem by introducing a new system service aimed at such a developer, who can now draw on standard tools and easily implement systems having this sort of structure.

Although the Astrolabe software was developed outside of Cornell, with professional coding standards and quality assurance, the overall approach reflects our Cornell work, the system is available for use at Cornell, and we have been exploiting the Astrolabe system as a platform in many of our projects.

Impact: A successful but highly decentralized (server-less) technology for securely monitoring, controlling and administering large systems and networked applications has the potential to revolutionize the way that applications are constructed and to take reliability and security to a completely new level. Such a technology would allow developers to automate administrative tasks now requiring human intervention and, by so doing, reduce the staffing needed to deploy and operate a complex system. Cost of ownership is one of the primary metrics used by the military and by commercial installations to determine the feasibility of new projects – lower COI (Cost Of Investment) can open the door to revolutionary advances in productivity and lead to the deployment of vital new technical capabilities. IBM's Autonomic Computing initiative revolves around these basic themes.

Our Opportunity and Role: Spinglass technologies use peer-to-peer gossip protocols that can be configured to run with no servers at all, yet one can create a kind of "virtual" server. Think of the way that a jigsaw puzzle is assembled by a team of players. Spinglass lets each computer in the system maintain some pieces of a shared puzzle – a shared system state object – and replicas of some of the other pieces. Our peer-to-peer protocols have the effect of assembling the data, on the fly, creating what look like copies of parts of the puzzle. The participants can behave as if the system had a single shared server that can respond to queries against this virtual data structure. The approach resulted in the development of the Astrolabe system. Astrolabe is a completely new kind of peer-to-peer system for gathering information at diverse sources, structuring it to resemble a hierarchical database, and then supporting decentralized queries and actions against that database. The system is flexible and can be customized with new data mining (aggregation) tasks at runtime, scalable (it imposes little load and can run on tens of thousands of nodes while maintaining worst-case delays in the seconds or minutes), and robust against disruptions.

Specific Accomplishments: Although we developed the Astrolabe system outside of Cornell, we evaluated it through simulation and experimental studies at Cornell, have made it available to others, and have been investigating extensions and security issues. Our initial Astrolabe implementation works extremely well, and is one of the central components of the proposed Air Force JBI architecture. We have optimized the system and obtained excellent performance, integrated it with firewalls and network address translators, interfaced it to a diversity of operating system platforms, and provide compatibility with Web Services technologies such as the ODBC/JDBC database interface standards and XML data encoding. However, the story is far from finished at this time. Although we have found ways to secure many aspects of the Astrolabe infrastructure, Astrolabe is not able to configure itself in a completely automatic manner and we have not been as successful securing the aggregation mechanisms, which could be disrupted by intruders. Moreover, Astrolabe needs more work to be used in settings with large numbers of wireless mobile nodes, or in settings where many machines might independently pose aggregation (data mining) queries. More work on these topics, and especially the security issues, will be needed.

Transitioning opportunities are also being explored. We have been in dialog with IBM about use of Astrolabe in Autonomic Computing applications emerging from its Web Services division, and used an Astrolabe-like solution in work we undertook jointly with Microsoft on a new generation of data center and clustering management systems. The Air Force JBI represents a good transitioning opportunity for us, but we are also exploring dialog with other military branches and with Homeland Defense. A key to success will be to actually demonstrate the value of Astrolabe in settings where standard technologies used in standard ways have serious limitations or scalability problems. We are currently undertaking several rather practical development efforts with this in mind – one in the context of the JBI publish-subscribe API mentioned earlier, and the other focused on administration and control of large electric power grids (for this, we are working with a team of electric power researchers associated with PSERC, a consortium funded by the power systems industry and charged with developing new solutions for the restructured power grid).

We have published extensively on Astrolabe [1, 6, 8, 9, 10, 11, 16, 20, 27, 28, 29, 31, 32, 33, 35 and 37]. We recommend [28] as the primary reference; this paper appeared in ACM Transactions on Computer Systems in May, 2003. Ken Birman has been asked to give several talks on Astrolabe in Autonomic Computing events and also spoke on this subject at Berkeley's CITRIS institute, where possible applications to critical infrastructure protection are under study. Examples of publications that focused on the use of Astrolabe in critical infrastructure settings include [10, 19, 28, 44].

IV. KELIPS.

Context: In a large distributed system, information is often out there but knowing where to look for it can be a huge challenge. In a military situation, the soldier in the field may be looking for resources – ammunition, food, water, and medical help. Knowing where to find such items can be crucial to carrying out a mission. In a hospital, one might want to find the display device currently closest to a physician so as to display the results of a medical test. Today we lack scalable, secure, high performance solutions to such problems. The usual way of solving them is to put the location data on a server somewhere and update it as things change. But this makes the server a central point of failure for the whole system, and also scales poorly.

Impact: Better solutions to such problems could enable applications with the sorts of behaviors just described, but it would be an exaggeration to say that distributed indexing is as important as scalable message delivery or distributed state monitoring. A better indexing solution could emerge as a valuable part of a set of solutions for building better large-scale systems.

Our Opportunity and Role: We used our peer-to-peer technologies to build the world’s most scalable and highest performance distributed indexing system. We call the resulting system Kelips (named for a type of self-synchronizing firefly native to Indonesia).

Specific Accomplishments: We implemented Kelips and also built a trace-driven simulation harness with which our implementation can be subjected to various realistic environments and conditions [22]. Kelips achieves O(1-hop) lookup delays compared to O(log N) delays in other such systems – an important advance in settings where the log might be a number like 20, and a “hop” might cost as much as 100ms. Storage costs are low and Kelips is far more robust when a system experiences high rates of join, leave, or failure [2]. On the other hand, Kelips handles very high update rates less efficiently than some of the indexing systems with slower lookups, so there are cost tradeoffs involved. We also need to invest more effort in the security issues associated with these kinds of systems, which are currently poorly understood.

V. WILLOW

Context: Bimodal Multicast requires an unreliable multicast routing protocol for the initial dissemination of messages to all recipients. While IP multicast sounds ideally suited for this purpose, it is unfortunately badly supported in today's Internet. The problem is that IP multicast addresses do not aggregate like point-to-point addresses do, and are therefore likely to fill routing tables if widely deployed and used. Many research groups have been looking at building multicast trees over point-to-point connections between end-hosts. Such solutions do not require modifications to Internet routers. However, building and maintaining efficient multicast trees is difficult, particular in the face of network and application dynamics.

Impact: A scalable, self-configuring, self-repairing, and location-aware application-level multicast routing facility could provide the underlying routing facility for many important collaborative protocols, including bimodal multicast and publish/subscribe and replace the need for an IP multicast infrastructure, which is currently non-existent.

Our Opportunity and Role: We are highly knowledgeable of collaborative applications and protocols, and thus very aware of their routing needs. We also have extensive experience with working with various networks, ranging from excellent to flaky.

Specific Accomplishments: We started with a multicast routing protocol called "SelectCast" that exploited Astrolabe [12, 21]. Astrolabe was used to find which regions contained multicast subscribers, and then SelectCast routed messages to those regions. While this worked reasonably well, we had little control over the tree branching factor this way, and the resulting tree was not necessarily ideally balanced. Also, because there was only one tree, the generated load across the members was necessarily uneven.

We then developed Willow, a novel peer-to-peer protocol that has Astrolabe-like functionality buried in it in order to determine where to route messages (publication was not completed during the period covered by this report). Unlike Astrolabe, however, Willow discovers and uses network locality information automatically in order to build close-to-optimal multicast dissemination trees. Such trees place nearby nodes close to one another, and use a higher branching factor near the root of the tree than near the leaves. Also, rather than having a single tree, Willow builds many trees in order to spread the load evenly across the members. Willow has an extremely small footprint that will allow it to be used virtually anywhere; possibly including ad hoc routing networks.

At this point in time, only a prototype Java implementation exists, but which can already demonstrate multicast dissemination and aggregation facilities quite well. More work is required though to turn it into a mature routing protocol.

Our student, Adrian Bozdog who graduated recently, also developed software to make Willow fully compatible with IP multicast. It exploits hardware multicast within sites, and uses Willow to route messages between sites. This routing software was extensively tested on Emulab, and proves to be very efficient for many types of multicast applications.

VI. BUILDING BETTER WIRELESS INFORMATION SYSTEMS.

Context: A second theme of our work has been to understand how wireless communication can be supported more seamlessly and more securely. Current mobility solutions are awkward, insecure, and unreliable. We believe that the Spinglass technologies have considerable potential to impact this situation.

Impact: Many kinds of military computing systems are mobile and use wireless links. Indeed, the Army is well advanced on a project to put all Army computing systems onto a military standard network based on the Internet. The Air Force has the challenge of communication from aircraft moving at high speeds and sometimes performing evasive maneuvers or operating in jammed environments. Even the Navy now favors wireless networks – on a battleship, wires are bulky and hard to install. Wireless systems take up less space. Better technologies for communicating with mobile platforms will enable the mobile combatant to maintain better information such as maps and other targeting resources, while also letting mobile systems better exploit the connectivity available to them.

Our Opportunity and Role: While studying issues arising from mobile use of Astrolabe, we realized that there was a need for better wireless communications support of a more basic type. Accordingly, we worked to develop new wireless routing protocols (particularly for "ad-hoc" networks where the mobile devices assist one-another in routing data back to the command post), a new TCP-like protocol that can support priorities and deadlines (e.g. routine updates to a map might be lower priority than updates reflecting new information on SAM sites), a new mobile file system that runs on this protocol, and (still under development) a new mobile Astrolabe-like system for tracking local information while moving around in a complex mobile environment.

Specific Accomplishments: Ben Atkin, a graduate student in the group, developed a protocol we call ATP: Adaptive Transport Protocol (ATP) [23, 35]. This protocol is similar to the well-known Internet TCP protocol, and in fact can even run over TCP. However, whereas a conventional TCP link lacks specific mechanisms for mobile settings, ATP provides them. The additional features include priorities, deadlines, better handling of dynamically changing connectivity and bandwidth, and interfaces to assist the application in adapting to a conditions change.

In brief, Atkin found that when file-access applications (file systems or web browsers) run over TCP to the mobile device, the handling of communication dynamics can be very poor [11]. TCP congestion control was designed for the Internet and may "kick in" inappropriately (Hari Balikrishnan of MIT first pointed this out and proposed a solution, but that solution has not been adopted hence commercial wireless devices need to explore other options). Moreover, when running a priority-based mechanism over multiple TCP channels, the channels can interfere with one-another in undesired ways. Ben's work, which introduces both priorities and deadlines, shows that ATP is able to give very good performance when layered over TCP, even in situations where conventional use of TCP breaks down. When layered over UDP, Ben's protocol does even better.

For his doctoral thesis, Atkin focused on developing applications that exploit the ATP interfaces. He concluded that the most fruitful direction involves mobile file systems for support of collaboration applications and groupware [11].

Our research also had an unexpected spinoff: it led to a completely new way of employing PCMCIA network cards in mobile settings, and Microsoft recently decided to offer this as a free feature of a future release of Windows XP. The key idea is to time-share one card across multiple ad-hoc and infrastructure networks, saving power and reducing weight while giving the user better connectivity; this may sound simple, but is actually tricky because when a card moves from network to network, the machine is effectively inaccessible on the networks to which it is no longer bound, and protocols can be disrupted by such behavior unless care is taken to hide the phenomenon from the endpoints. This transition is of special interest because Microsoft won't charge for the technology. Thus, a solution of military value will soon be available, for no fee of any kind, to the military on the platform it uses most widely. If the technology proves valuable, it will save power and weight (just one PCMCIA card per laptop instead of several), money (there is no need to purchase those extra cards, or software to network them), and time to field new solutions (since the technology will become part of the broad commercial base and hence easily used and well supported). Taken jointly, these represent significant costs – indeed, it is entirely possible that this one transition will save the military more money than was spent to support our entire FTN research effort! We believe that as our other Spinglass technologies transition, the value will be orders of magnitude higher, but these kinds of events do point to the directly measurable value of research at organizations such as DARPA.

VII. UNDERSTANDING NATIONALLY CRITICAL INFRASTRUCTURE NEEDS AND VALIDATING OUR SOLUTIONS.

Context: Presidents Clinton and Bush have emphasized that work is needed on improving the robustness of nationally critical infrastructure. Control of the restructured electric power grid is cited as a top priority. DARPA has cited critical infrastructure defense as an important priority in many studies, including the 1995 ISAT study on Survivability of Critical Infrastructure, an effort on which Birman participated.

Impact: Improving the security and robustness of nationally critical infrastructure could ward off future terrorist attacks and also reduce the chances that routine mishaps (storms, earthquakes, accidental cutting of cables) might disable major parts of a nationally critical resource such as the electric power grid, telecommunications network, Internet, air traffic control system, etc.

Our Opportunity and Role: We started a dialog with a team associated with PSERC, a major research center funded in part by the Electric Power Research Industry Consortium, EPRI. Our work focuses on identifying new kinds of grid monitoring and control requirements stemming from restructuring and deregulation, showing how our tools can be adapted to solve such problems, and evaluating the solutions using high-fidelity simulations developed by EPRI and considered to be convincing by domain experts. Much of the funding for this work was actually provided by NSF on a grant unrelated to our DARPA funding, but the software tools we used (Astrolabe, Bimodal Multicast and Gravitational Gossip, etc) were developed under DARPA funding on this FTN grant.

Specific Accomplishments: Gradate student Ken Hopkinson has taken the lead on this effort. As we finalize his report, he continues to enlarge his work on a novel simulation system that links a widely used simulator for the electric power grid to the most powerful network protocol simulator currently available. Hopkinson has been using and validating his basic simulator while also extending it by connecting it to a larger-scale simulation technology similar to the small academic version on which his initial work was done. The resulting “mixed mode” simulator permits us to simulate new control strategies and new network protocols for the restructured electric power grid with a degree of realism never before achieved [10, 19, 28, 44]. For this work we teamed with a national center, PSERC, that brings together leading researchers from the electric power research community with others interested in the fundamental challenges of this rapidly changing but nationally critical infrastructure area.

Specific findings, to date, include the surprising discovery that the standard EPRI architecture for network control of power grids has a serious weakness [10, 19]. We simulated standard grid protection algorithms as protocols running on the EPRI architecture and discovered that TCP flow and congestion control policies cause such serious disruption that the protection scheme will often malfunction. When we substitute an Astrolabe-based monitoring mechanism, using

Gravitational Gossip for notification, we believe that this problem can be completely overcome. Hopkinson is now working to demonstrate this result and, if successful, we believe it could pave the way for widespread use of Astrolabe within the electric power industry [10].

Hopkinson joined the Air Force Institute of Technology (AFIT) in Dayton, Ohio as an Assistant Professor in Fall 04.

VIII. COMPOSITIONAL PROOF TECHNIQUES FOR SCALABLE SYSTEMS.

Context: Commercial networking software is too unreliable and too insecure to be used in safety-critical applications, especially in the military. Formal proof techniques provide an adequate science base for building systems to control critical software infrastructure, but the current techniques do not scale well.

Impact: We are creating highly innovative proof techniques and systems that can substantially improve the reliability, adaptability, and performance of networking software. We have built a theorem proving system called a Logical Programming Environment (LPE) and used it to formally specify and check properties of system design and code as it is being developed, as well as to verify and optimize code that has already been written. Publications on the work include [48, 53, 57, 69, 73, 76].

Our Opportunity and Role: Our theorem proving work on Ensemble and Spinglass has resulted in major advances in the science and technology for building provably correct computing systems from small, compositional components. Component architectures are widely used in industry and this accomplishment means that for the first time, one can talk about a ``push-button" methodology for automatically generating correct components to control sensor networks and other complex hardware systems containing large numbers of small computers [48, 53]. We have enlarged the scope of coverage beyond the kinds of protocols used in Ensemble to include other kinds of networking protocols such as the automatic generation of coordinated contracts for real-time networks. Our work also led to significant extensions to the LPE's logical foundations and its automated reasoning capabilities [57, 59, 73, 76].

Specific Accomplishments:

Optimization of Communication Systems: Using the LPE, we have developed fully automatic, semantics-based tools for improving the code of the Ensemble group communication system [53]. The tools create the code of a fast-path through the protocol stack and integrate it into the Ensemble system [53, 57, 73]. The improved code operates three to ten times faster than the original and is generated in a matter of seconds [76]. Comparable improvements done by hand took months of tedious and complex work on smaller examples, and the complexity led to errors in the faster code. In contrast, the code modifications created by the automatic tools are formally proven to be correct, that is, the improved code computes the exact same results as the original.

Compositional Verification of Protocol Stacks: Using a modified version of formal IO-automata we have implemented a method for decomposing the task of verifying the properties of protocols and protocol stacks into small reusable components. IO-automata represent protocols, stacks, and groups of stacks and our method allows us to compose these automata in an elegant fashion such that the composed automaton inherits almost all properties from its individual components. This makes it possible to verify properties of systems in a modular and incremental way.

We have applied this method to the verification of Ensemble's total order protocol and later to the design and verification of new adaptive protocols in Ensemble [76]. We have created a basic library of facts about Ensemble that are formally stated and proved. These serve as the basis of specification of problems and verification of designs.

Formal Design of Adaptive Systems*:* We have designed a generic switching protocol for the construction of adaptive network systems and formally proved it correct with the Logical Programming Environment. In the process we have developed a formal characterization of communication properties that can be preserved when the system switches between different protocols. We have also developed an abstract characterization of invariants that have to be satisfied by an implementation of the switching protocol in order to work correctly.

As foundation for this work we have introduced the novel concept of meta-properties. Meta-properties make it possible to give an abstract characterization of "switchable" system properties, which in turn makes it easier to check whether a specific set of protocols can be employed in an adaptive system. We have described switchable properties in terms of several meta-properties such as "safety", "asynchrony", "delayable", and "send-enabled", as well as "composability" and "memorylessness". The first four of these properties are required for any layered communication system while the latter are necessary for switching. The abstract approach represents a major increase in our formal understanding of distributed systems and makes it possible to support the formal analysis and design of networked systems.

With the LPE we have formally proven that communication properties that satisfy these six meta-properties are preserved under switching, whenever the switch maintains a simple synchronization invariant. The verification efforts revealed a variety of implicit assumptions that are usually made when designing communication systems and uncovered minor design errors that would have otherwise made their way into the implementation.

We have evaluated the performance implications of using our hybrid protocol by switching between two well-known mechanisms for implementing total order and shown that switching close to the cross-over point of these protocols performance leads to the best practical results.

Advanced Reasoning Capabilities*:* We have significantly enhanced the automatic reasoning tools of the LPE by adding generic proof techniques that support the verification of networked systems and their implementations and proof strategies especially tailored towards reasoning about program composition, aspect weaving, and embedded systems. Substantial new reasoning capabilities are now in place.

We have integrated JProver, a fully automated theorem prover for constructive first-order logic, as an external proof engine into the LPE. JProver operates on matrices and connections, a very compact representation of the search space that substantially reduces the time needed for finding proofs. Extensions of Jprover towards inductive theorem proving have been explored in theory and are currently being added to the theorem prover.

We have introduced new techniques for asynchronous and parallel theorem proving and are currently adding strategies that utilize external proof systems such as PVS and MetaPRL as well as constraint solvers and computer algebra systems.

We have implemented tools that enable the verification system to learn from the work we have already done by "mining" proofs for reasoning steps that can be reused as "derived inference rules".

The use of these techniques has significantly increased the degree of automation in formal design and verification and will increase the productivity of rigorous design methods.

ROLES IN STANDARDS AND INDUSTRY INITIATIVES

Werner Vogels has participated in the standards group formalizing standards for Web Services. His work on the WS_MEMBERSHIP service has been proposed as a standard, and he is now working with a team exploring group mechanisms and fault-tolerance [16, 17].

Ken Birman has worked with IBM on the Autonomic Computing initiative. He was a speaker at the Almaden workshop on Autonomic Computing and is now in dialog about application of Astrolabe to challenges encountered in IBM's WebSphere product line [8]. Birman gave keynote talks at three IBM workshops on Autonomic Computing.

Robbert Van Renesse has been an active participant in the Global Grid Computing Forum and has been exploring applications of Astrolabe to Global Grid Computing.

EDUCATIONAL CONTRIBUTIONS FROM OUR EFFORT

A research group such as ours makes multiple kinds of contributions. Over the period of the grant, these include the following:

- Research papers, including a new textbook ("Reliable Distributed Systems," to be published by Springer Verlag in January 2005).

- Software. As noted earlier, we make all our Cornell-developed software available, for free, to researchers and developers worldwide. However, we do not provide support (the costs of doing so are prohibitive in a non-corporate setting), and we recognize that academically developed software can be less robust than professionally developed systems. In particular, many academic groups worldwide use our Cornell software in their own educational programs, as the basis for large student projects. We view this as a very important way of "amplifying" our work.

- New academic curricula. At Cornell, we've made "information assurance" a central part of our academic program, and the DARPA funding has played a central role in establishing the context in which we were able to do this. The textbook mentioned above, "Reliable Distributed Systems", emerged from such a course. Cornell's syllabi and materials for these courses are available to others and we often have visitors for a week or more who come to learn how we teach this material.

- Students. As noted before, one of our PhD students (Ken Hopkinson, who worked on applications of Astrolabe in the Electric Power Grid) will take a position at the Air Force Institute of Technology in Dayton, Ohio, as an Assistant Professor. But this is just the most dramatic of a long string of less dramatic but equally important accomplishments. We train students at every level (undergraduate, Masters and PhD) in the concepts and technology of security and reliability. And these students often take roles in industry or government that let them continue their work on these topics. Over the four years of FTN and SRS seedling support, Cornell has seen perhaps 1000 students graduate (in total), consisting of some 550 undergraduates, 400 Masters students and 50 PhDs. *Every one of these students has a deep, hands-on background in security, fault-tolerant distributed computing, and other aspects of information assurance.*

As an interesting aside, we note that while many of our students are foreign, the majority take positions here in the United States. We track the careers of as many of our students as possible and find that irrespective of their national origin, 95% or more remain in America, and indeed the vast majority eventually become U.S. permanent residents and ultimately citizens.

PUBLICATIONS

This lists all publications by our group for which this DARPA grant supported our work. A complete list of publications over the entire period (including work not supported by this grant) is available at http://www.cs.cornell.edu/Info/Projects/Spinglass/pubs.html.

2004

1. **Scalable, Self-Organizing Technology for Sensor Networks.** Kenneth P. Birman, Saikat Guha, Rohan Murty. *Advances in Pervasive Computing and Networking*, Bulent Yeler, ed. Kluwer Academic Press, Fall 2004.
2. **Kache: Peer-to-Peer Web Caching Using Kelips**. Prakash Linga, Indranil Gupta, and Ken Birman. Submitted to ACM Transactions on Information Systems (TOIS), June 2004
3. "**On the Placement of Internet Taps in Wireless Neighborhood Networks**", Ranveer Chandra, Kamal Jain, Mohammad Mahdian and Lili Qiu. In Submission.
4. "**MultiNet: Connecting to Multiple IEEE 802.11 Networks Using a Single Wireless Card**", Ranveer Chandra, Paramvir Bahl and Pradeep Bahl. *IEEE Infocom*, March 2004 Hong Kong.
5. **Like it or not, Web Services *are* Distributed Objects!** K.P. Birman, Comm. of the ACM, *Viewpoints Column.* May or June 2004.
6. **The Performance of SelectCast - Scalable and Self-Repairing Multicast Overlay Routing**. Adrian Bozdog, Robbert van Renesse, Dan Dumitriu. Submitted to the special SPE issue on "Experiences with Auto-adaptive and Reconfigurable Systems".
7. **Practical algorithms for Size estimation in Large and Dynamic groups**. D. Psaltoulis, D. Kostoulas, I. Gupta, K. Birman, A. Demers. Submitted to: Twenty-Third Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC 2004), July 25-28, 2004, New Foundland, Canada.
8. **Adding High Availability and Autonomic Behavior to Web Services.** Ken Birman, Robbert van Renesse, Werner Vogels. In the Proceedings of the 26th Annual International Conference on Software Engineering (ICSE 2004). May 23 - 28, 2004. Edinburgh, Scotland.
9. **Building Scalable Solutions to Distributed Computing Problems using Probabilistic Components.** Kenneth P. Birman and Indranil Gupta. Cornell Technical Report.

2003

10. **Overcoming Communications Challenges in Software for Monitoring and Controlling Power Systems.** Kenneth P. Birman, Jie Chen, Ken Hopkinson, Bob Thomas, Jim Thorp, Robbert van Renesse, Werner Vogels. Submitted to special issue of the Proceedings of the IEEE, "Energy Infrastructure Defense Systems", October 2003

11. **MFS: an Adaptive Distributed File System for Mobile Hosts.** Benjamin Atkin and Kenneth P. Birman. Cornell University Technical Report

12. **SelectCast -- A Scalable and Self-Repairing Multicast Overlay Routing Facility**. Adrian Bozdog, Robbert van Renesse, Dan Dumitriu. Proceedings of the First ACM Workshop on Survivable and Self-Regenerative Systems. October 31, 2003. Fairfax, VA.

13. **The League of SuperNets**, Ken Birman, IEEE Internet Computing, vol. 7, no. 5, 2003, pp. 92-96.

14. Architectural Challenges for Global Information Systems. Werner Vogels. In the Proceedings of the High Performance Transaction Systems Workshop, Asilomar, CA, October 2003.

15. **Benchmarking CLI for High Performance Computing.** Werner Vogels, Submitted to IEEE Software Special Edition on SSCLI technology.

16. **Tracking service availability in long running business activities**. Werner Vogels. Submitted to The First International Conference on Service Oriented Computing (ICSOC03).

17. **WS-Membership - Failure Management in a Web-Services World.** Werner Vogels , Chris Re. 12th International World Wide Web Conference , May 2003 (Budapest, Hungary).

18. **A Churn-Resistant Peer-to-Peer Web Caching System**. Prakash Linga, Indranil Gupta, Ken Birman. ACM Workshop on Survivable and Self-Regenerative Systems, October 2003.

19. **EPOCHS: Integrated Cots Software For Agent-Based Electric Power And Communication Simulation** Hopkinson, K.M.; Giovanini, R.; Wang, X.; Birman, K.P.; Coury, D.V.;Thorp, J.S., EPOCHS: Integrated COTS Software for Agent-based Electric Power and Communication Simulation.2003 Winter Simulation Conference.7-10 of December 2003, New Orleans, USA.

20. **Navigating in the Storm: Using Astrolabe for Distributed Self-Configuration, Monitoring and Adaptation** Ken Birman , Robbert van Renesse, Werner Vogels 5th Annual International Active Middleware Workshop (AMS 2003), Seattle , WA June 2003. *also Submitted to the special issue of Cluster Computing on "Autonomic Computing", June 2003*

21. **Heterogeneity-Aware Peer-to-Peer Multicast.** Robbert van Renesse, Ken Birman, Adrian Bozdog, Dan Dumitriu, Manpreet Singh and Werner Vogels. Proceedings of the17th International Symposium on Distributed Computing (DISC 2003). October 2003. Sorrento, Italy

22. **Kelips: Building an Efficient and Stable P2P DHT Through Increased Memory and Background Overhead.** Indranil Gupta, Ken Birman, Prakash Linga, Al Demers and Robbert van Renesse. Submitted to: 2nd International Workshop on Peer-to-Peer Systems (IPTPS '03); February 20-21, 2003. Claremont Hotel, Berkeley, CA, USA.

23. **Evaluation of an Adaptive Transport Protocol.** Benjamin Atkin and Kenneth P. Birman. ACM INFOCOM 2003, April 1-3 2003, San Francisco.

24. **User-specified Adaptive Scheduling in a Streaming Media Network**. Michael Hicks, Adithya Nagarjan, Robbert van Renesse. Proc. of OpenARCH'03, San Francisco, CA. April 2003

25. **Web Services are not Distributed Objects**. Werner Vogels, IEEE Internet Computing, Vol. 7, No. 6, pp 59-66, November/December 2003. Online at http://weblogs.cs.cornell.edu/AllThingsDistributed/archives/000343.html

26. **WAIF: Web of Asynchronous Information Filters** Dag Johansen, Robbert van Renesse, and Fred B. Schneider. Springer-Verlag Lecture Notes in Computer Science volume 2584 "Future Directions in Distributed Computing". A. Schiper, A. A. Shvartsman, H. Weatherspoon, and B. Y. Zhao, editors. Springer-Verlag, Heidelberg, April 2003

27. **The Importance of Aggregation**. Robbert van Renesse. Springer-Verlag Lecture Notes in Computer Science volume 2584 "Future Directions in Distributed Computing". A. Schiper, A. A. Shvartsman, H. Weatherspoon, and B. Y. Zhao, editors. Springer-Verlag, Heidelberg, April 2003

28. **Astrolabe: A Robust and Scalable Technology for Distributed System Monitoring, Management, and Data Mining.** Robbert van Renesse, Kenneth Birman and Werner Vogels. ACM Transactions on Computer Systems, May 2003, Vol.21, No. 2, pp 164-206

2002

29. **The Power of Epidemics: Robust Communication for Large-Scale Distributed Systems**. Werner Vogels, Robbert van Renesse and Ken Birman. In Proceedings of HotNets-I '02: First Workshop on Hot Topics in Networks, special issue of the ACM SIGCOMM Computer Communication Review, Princeton, NJ. October 2002.

30. **Power-Aware Epidemics.** Robbert van Renesse. In Proceedings of the International Workshop on Reliable Peer-to-Peer Systems, Osaka, Japan. October 2002.

31. **Astrolabe: A Robust and Scalable Technology for Distributed System Monitoring, Management, and Data Mining.** Robbert van Renesse, Kenneth Birman and Werner

Vogels. To appear in May 2003, ACM Transactions on Computer Systems (TOCS), TR first published November 2001 (Revised September 2002)

32. **The Surprising Power of Epidemic Communication.** Kenneth Birman. Proceedings, Workshop on Future Directions in Distributed Computing (FuDiCo 2002). Bertinoro, Italy (June 2002). Springer-Verlag.

33. **The Importance of Aggregation.** Robbert van Renesse. In Proceedings of the International Workshop on Future Directions in Distributed Computing, Bertinoro, Italy. June 2002.

34. **Holistic Operations in Large-Scale Sensor Network Systems: a Probabilistic Peer-to-Peer Approach.** Indranil Gupta & Kenneth Birman. In Proceedings, International Workshop on Future Directions in Distributed Computing (FuDiCo). June 2002. pp. 1-4.

35. **Fighting Fire with Fire: Using Randomized Gossip to Combat Stochastic Scalability Limits**. Indranil Gupta Kenneth P. Birman and Robbert van Renesse. (ed. Nong Ye) Special Issue of Quality and Reliability of Computer Network Systems, Journal of Quality and Reliability Engineering International, May/June 2002, Vol. 18, No. 3, pp 165-184.

36. **TAF: A Temporal Adaptive Framework for Hybrid Routing in Mobile Ad Hoc Networks.** Venugopalan Ramasubramanian and Emin Gun Sirer. Technical Report, TR2000-1862, Department of Computer Science, Cornell University. March 2002.

37. **Scalable Management and Data Mining Using Astrolabe**. van Renesse, Robbert, Birman, Kenneth P., Dumitriu, Dan and Vogel, Werner. Proceedings of the First International Workshop on Peer-to-Peer Systems (IPTPS). Cambridge, Massachusetts. March 2002.

38. **A Collaborative Infrastructure for Scalable and Robust News Delivery.** Werner Vogels, Chris Re, Robbert van Renesse and Ken Birman. In the Proceedings of the IEEE Workshop on Resource Sharing in Massively Distributed Systems (RESH'02), Vienna, Austria, July 2002.

39. **Scalable Data Fusion Using Astrolabe**. Ken Birman, Robbert van Renesse and Werner Vogels. In proceedings of the Fifth International Conference on Information Fusion 2002 (IF 2002), July 2002.

40. **Collaborative Content Delivery: A Peer-to-Peer Solution for Web-Based Publish/Subscribe**. Werner Vogels, Robbert van Renesse, Ken Birman. Submitted to the First International Workshop on Peer-to-Peer Systems (IPTPS 2002).

41. **Optimizing Buffer Management for Reliable Multicast**. Zhen Xiao, Robbert van Renesse, Kenneth Birman. Proceedings of the International Conference on Dependable Systems and Networks (DSN ’02), June 2002.

42. **SWIM: Scalable Weakly-consistent Infection-style Process Group Membership**

Protocol. Abhinandan Das, Indranil Gupta, Ashish Motivala. In Proceedings of the International Conference on Dependable Systems and Networks 2002 (DSN 2002), pp 303-312. June 2002.

43. **Efficient Epidemic-Style Protocols for Reliable and Scalable Multicast.** Indranil Gupta, Anne-Marie Kermarrec, Ayalvadi J. Ganesh. Proceedings of the 21st Symposium on Reliable Distributed Systems (SRDS 02), Osaka, Japan. October 2002. pp. 180-189.
44. **An Agent-based Current Differential Relay for use with a Utility Intranet**. D.V. Coury, J.S. Thorp, K.M. Hopkinson, K.P. Birman. IEEE Transactions on Power Delivery, January 22, 2002, Vol. 17, No 1, pp 47-53.
45. **Mobility Aware Adaptation in the Transport Protocol.** Benjamin Atkin and Ken Birman. Submitted to ACM SIGCOMM 2002. (January 2002)

2001

46. **The Architecture and Performance of the Security Protocols in the Ensemble Group Communication System.** Ohad Rodeh, Ken Birman, Danny Dolev. Journal of ACM Transactions on Information Systems and Security (TISSEC).
47. **Using AVL Trees for Fault-Tolerant Group Key Management**. Ohad Rodeh, Kenneth P. Birman, Danny Dolev. International Journal of Information Security (IJIS), Vol. 1, No 2, pp 84-99, February 2002; Published online: October 26, 2001.
48. **Proving Hybrid Protocols Correct, Mark Bickford.** Christoph Kreitz, Robbert van Renesse, Xiaoming Liu. In the International Conference on Theorem Proving and Higher Order Logic 2001. Nijmegen, The Netherlands. August 2001.
49. **On Scalable and Efficient Distributed Failure Detectors.** Indranil Gupta, Tushar D. Chandra, German Goldszmidt. In 20th Symposium on Principles of Distributed Computing (PODC 2001), pp. 170-179, Newport, RI, August, 2001.
50. **Scalable Fault-tolerant Aggregation in Large Process Groups.** Indranil Gupta, Robbert van Renesse, Kenneth P. Birman. International Conference on Dependable Systems and Networks (DSN '01) Gothenberg, Sweden, July 2001.
51. **Technology Requirements for Virtual Overlay Networks**. Ken Birman. IEEE Systems, Man and Cybernetics: Special issue on Information Assurance, Vol. 31, No 4, pp 319-327, July 2001.
52. **Spinglass: Secure and Scalable Communications Tools for Mission-Critical Computing.** Kenneth P. Birman, Robbert van Renesse and Werner Vogels. International Survivability Conference and Exposition. DARPA DISCEX-2001, Anaheim, California, June 2001.
53. **An Experiment in Formal Design using Meta-Properties.** Mark Bickford, Christoph Kreitz, Robbert van Renesse, Robert Constable. In the DARPA Information

Survivability Conference and Exposition II (DISCEX 2001), IEEE Computer Society Press. June 2001

54. **Fighting Fire with Fire: Using Probabilistic Protocols to Overcome Stochastic Network Failures.** Ken Birman. Submitted to: ACM SIGOPS Symposium on Operating System Principles (SOSP-18), March 2001.

55. **A Gossip Protocol for Subgroup Multicast.** Kate Jenkins, Ken Hopkins and Ken Birman. International Workshop on Applied Reliable Group Communication (WARGC 2001), Phoenix, Arizona, April 2001.

56. **Providing Efficient, Robust Error Recovery Through Randomization**. Zhen Xiao and Ken Birman. International Workshop on Applied Reliable Group Communication (WARGC 2001), Phoenix, Arizona, April 2001.

57. **Protocol Switching: Exploiting Meta Properties**. Xiaoming Liu, Robbert van Renesse, Mark Bickford, Christoph Kreitz, Robert Constable. In the International Workshop on Applied Reliable Group Communication at the International Conference on Distributed Computing Systems (ICDCS), Phoenix, AZ, April 2001.

58. **Anonymous Gossip: Improving Multicast Reliability in Ad-Hoc Networks.** Ranveer Chandra, Venugopalan Ramasubramanian, Ken Birman. International Conference on Distributed Computing Systems (ICDCS 2001), Phoenix, Arizona, April 2001.

59. **A Randomized Error Recovery Algorithm for Reliable Multicast.** Zhen Xiao and Ken Birman. IEEE Infocom 2001, April 2001, Alaska.

60. **Using Epidemic Techniques for Building Ultra-Scalable Reliable Communications Systems.** Werner Vogels, Robbert van Renesse, and Ken Birman. Workshop on New visions for Large-Scale Networks: Research and Applications, Vienna, VA, March 2001.

2000

61. **An Overview of the Galaxy Management Framework for Scalable Enterprise Cluster Computing.** Werner Vogels and Dan Dumitriu. In the Proc. of the IEEE International Conference on Cluster Computing: Cluster-2000, Chemnitz, Germany, December 2000.

62. **A Probabilistically Correct Leader Election Protocol for Large Groups.** Indranil Gupta, Robbert Van Renesse, Ken Birman, DISC 2000, Toledo, Spain, October 4-6, 2000.

63. **Scalability, Throughput Stability and Efficient Buffering in Reliable Multicast Protocols.** Oznur Ozkasap. Technical Report, TR2000-1827, Department of Computer Science, Cornell University. December 2000.

64. **Next Generation Internet: Unsafe at Any Speed?** Ken Birman. IEEE Computer,

Special Issue on Infrastructure Protection, Vol. 33, No 8, pp 54-88, August 2000.

65. **Throughput Stability of Reliable Multicast Protocols.** Oznur Ozkasap, Ken Birman. ADVIS' 2000, Dokuz Eylul University, Izmir, Turkey, October 25-27, 2000.

66. **Design and Implementation of Programmable Media Gateways.** Wei Tsang Ooi, Robbert van Renesse, Brian Smith. In the 10th International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV 2000), Chapel Hill, NC. June 26-28, 2000.

67. **A Simulation Model for an Epidemic Multicast Protocol.** Oznur Ozkasap and Ken Birman. BAS2000 Conference (5th Computer Networks Symposium), Bilkent University, Ankark, Turkey, June 15-16, 2000.

68. **Technology Challenges for Virtual Overlay Networks.** Ken Birman. IEEE Systems, Man, and Cybernetics Information Assurance and Security Workshop, June 6-7, 2000, West Point, New York.

69. **Fast Protocol Transition in A Distributed Environment.** Xiaoming Liu, Robbert van Renesse. (short paper). In the Proc. of 19th ACM Conference on Principles of Distributed Computing (PODC 2000). Portland, OR, July 2000.

70. **A Study of Group Rekeying.** Ohad Rodeh, Ken Birman, Danny Dolev. Cornell University, Computer Science TR2000-1791, March 2000.

71. **Agent Technology Applied to Adaptive Relay Settings for Multi-Terminal Lines.** D.V. Coury, J.S. Thorp, K.M. Hopkinson, K.P. Birman, Cornell University, Computer Science TR2000-1792, March 2000. Submitted to the IEEE Summer Power Conference.

72. **Optimized Group Rekey for Group Communication Systems.** Ohad Rodeh, Ken Birman, and Danny Dolev, Network and Distributed System Security 2000, February 2000, San Diego, California. (Extended version available as Cornell University, Computer Science TR99-1764.)

73. **The Horus and Ensemble Projects: Accomplishments and Limitations.** Ken Birman, Robert Constable, Mark Hayden, Christopher Kreitz, Ohad Rodeh, Robbert van Renesse, Werner Vogels. Proc. of the DARPA Information Survivability Conference & Exposition (DISCEX '00), January 25-27 2000 in Hilton Head, South Carolina.

74. **An Adaptive Protocol for Locating Media Gateways.** Wei Tsang Ooi, Robbert van Renesse. In the 8th ACM International Multimedia Conference, Los Angeles, CA, 2000.

75. **Scalable and Secure Resource Location.** Robbert van Renesse. In Proc. of the 33rd Hawaii International Conference on System Sciences, Maui, Hawaii, January 2000.

1999

76. **Building Reliable, High-Performance Communication Systems from Components**. Xiaoming Liu, Christoph Kreitz, Robbert van Renesse, Jason Hickey, Mark Hayden, Ken Birman, and Robert Constable. In Proc. of the 17th ACM Symposium on Operating System Principles, Kiawah Island Resort, SC, December 1999.

77. **Efficient Buffering in Reliable Multicast Protocols.** Ozkasap, Oznur, van Renesse, Robbert, Birman, Kenneth and Xiao, Zhen Lecture notes in Computer Science 1736, Springer Verlag, pp 159-169. Proceedings of the First Workshop on Networked Group Communication.(NGC99) Pisa, Italy. (November 1999).

78. **Bimodal Multicast**. Kenneth P. Birman, Mark Hayden, Oznur Ozkasap, Zhen Xiao, Mihai Budiu and Yaron Minsky. ACM Transactions on Computer Systems, Vol. 17, No. 2, pp 41-88, Nov, 1999.

LIST OF ACRONYMS

ASTROLABE - Cornell’s scalable monitoring and data mining platform.
ATP – Adaptive Transport Protocol. A TCP-like protocol with QoS properties
BIMODAL MULTICAST
DHT – Distributed Hash Table
ENSEMBLE - Cornell’s penultimate group communication platform; widely used
EPIDEMIC - A biologically inspired communication protocol (mimics virus propagation)
GOSSIP - Style of communication used in Epidemic protocols
GIG - Global Information Grid. The primary DoD vision for information-centric warfare.
SOA - Service Oriented Architectures. Object oriented and Web Services systems.
HORUS - Cornell’s second group communication platform; set performance records
IP – Internet Protocol
ISIS - Cornell’s first group communication platform; widely used
JBI – Joint Battlespace Infosphere
JDBC – Java DataBase Connectivity
KELIPS – Cornell’s distributed “indexing” (lookup) platform
LAN - Local Area Network
LPE - Logical Programming Environment
NCES - Network Centric Enterprise Systems
NuPRL - Cornell-developed automated theorem prover and LPE
ODBC – Open DataBase Connectivity
PCMCIA - Personal Computer Memory Card International Association
QoS - Quality of Service (real-time guarantees, low-latency, low jitter, etc)
QUICKSILVER
SAN - Storage Area Network
SELECTCAST - A publish-subscribe capability build using Astrolabe
TCP – Transmission Control Protocol
UDP – User Datagram Protocol
VIRTUAL SYNCHRONY - Execution model used in Ensemble and Horus systems
WAN - Wide Area Network
WEB SERVICES - Popular architectural standard for distributed computing
WEBSPHERE - IBM Web Services platform product
WILLOW - Improved version of SelectCast