UNIVERSITY OF MACEDONIA

MASTER INFORMATION SYSTEMS

OVERLAY NETWORKS

Katsiani Maria

Networking Technologies
Professors: A.A. Economides &
A. Pomportsis

Thessaloniki, 10/01/2006
ΠΑΝΕΠΙΣΤΗΜΙΟ ΜΑΚΕΔΟΝΙΑΣ

ΠΜΣ ΠΛΗΡΟΦΟΡΙΑΚΑ ΣΥΣΤΗΜΑΤΑ

ΔΙΚΤΥΑ ΕΠΙΚΑΛΥΨΗΣ

Κατσιάνη Μαρία

Τεχνολογίες Τηλεπικοινωνιών & Δικτύων
Καθηγητές: Α.Α. Οικονομίδης & Α. Πομπόρτσης

Θεσσαλονίκη, 10/01/2006

ABSTRACT

In recent years a number of overlay networks have been developed. These systems aim to provide functionalities that cannot be easily or quickly provided by the IP network layer. For example, overlay networks have been proposed to provide ubiquitous multicast connectivity to end users, to provide robust end-to-end connectivity in the face of network outages, or to provide protection against DoS attacks [5]. Although overlay networks can provide new functionality and/or better performance to applications, their loose coupled nature raises new research challenges. One of the primary concerns in using an overlay network is the trustworthiness of its data delivery. Recently, overlay networks have emerged as a means to enhance end-to-end application performance and availability. Overlay networks attempt to leverage the inherent redundancy of the Internet's underlying routing infrastructure to detour packets along an alternate path when the given primary path becomes unavailable or suffers from congestion. However, the effectiveness of these overlay networks depends on the natural diversity of overlay paths between two endhosts in terms of physical links, routing infrastructure, administrative control, and geographical distribution.

Τα τελευταία χρόνια έχει αναπτυχθεί ένας μεγάλος αριθμός δικτύων επικάλυψης. Τα συστήματα αυτά έχουν ως σκοπό να παρέχουν λειτουργίες, οι οποίες δεν μπορούν εύκολα ή γρήγορα παραχθούν από το IP στρώμα δικτύου. Για παράδειγμα, τα δίκτυα επικάλυψης έχουν προταθεί για να παράγουν ευρέως διαδεδομένη συνδεσιμότητα στους τελικούς χρήστες, ή να παράγουν δυνατή και από άκρη σε άκρη σύνδεση , ή να παράγουν προστασία ενάντια σε επιθέσεις DoS. Παρόλο που τα δίκτυα επικάλυψης μπορούν να παράγουν νέες λειτουργίες και καλύτερη απόδοση σε εφαρμογές, η χαλαρή σύνδεση δημιουργεί περιθώριο για νέες έρευνες. Ένα από τα σημαντικά σημεία προσοχής κατά τη χρησιμοποίηση των δικτύων επικάλυψης είναι η αξιοπιστία στα πακέτα αποστολής. Πρόσφατα, τα δίκτυα επικάλυψης εμφανίστηκαν στο προσκήνιο ως ένα μέσο εμπλουτισμού των από άκρο σε άκρο εφαρμογών σε απόδοση και διαθεσιμότητα. Τα δίκτυα επικάλυψης επιχειρούν να ενισχύσουν την έμφυτη τάση για επανάληψη της βασικής υποδομής του διαδικτύου να αλλάζουν το δρομολόγιο των πακέτων, όταν η αρχική διαδρομή δεν είναι διαθέσιμη ή έχει υποστεί συμφόρηση. Παρόλα αυτά, η αποτελεσματικότητα των δικτύων αυτών εξαρτάται από τη ποικιλία της διαδρομής των δικτύων ανάμεσα στους δύο τελικούς χρήστες όσον αφορά τις φυσικές συνδέσεις, την επαναληπτική υποδομή, τον διοικητικό έλεγχο και τη γεωγραφική διανομή.

CONTENTS

ΠΕΡΙΕΧΟΜΕΝΑ

1. Introducton

Distributed applications would be well served by richer semantics than the network layer supplied by the Internet. Today's distributed applications have only one primitive from which they may build services: Internet Protocol (IP) Unicast, the best effort, single source and destination delivery of datagrams. Unicast delivery allows a

single network node to ask for a single packet of data to routed through the Internet to a single destination node. Additional semantics have been built on top of this single primitive. For example, Transmission Control Protocol (TCP) provides reliability and flow control.

Many applications, however, would benefit from additional services that are more difficult to build on top of IP Unicast delivery. Mission critical applications would like control over the way their packets are routed - perhaps trading off resource usage for reliability by using multi-path routing. Teleconferencing applications, chat rooms, and Internet broadcasting systems would benefit from efficient group communication. A stock ticker application might like to perform latency measurements over many paths to find a low latency path undetected by normal IP routing. A content distribution network would like to distribute and store data throughout the network.

One approach to addressing these needs is to build new network services into routers across the Internet. Generally this approach has two drawbacks. First, it may be inappropriate to add the necessary functions to routers that should remain fast and simple to ensure their continued availability as an important shared resource. Second, adding an important network service to routers is likely to support only those applications envisioned during the design of the service. For example, IP Multicast provides a single service model that is inappropriate for a number of multicasting applications. Efforts to revamp IP Multicast for reliability or for secure admission control require yet more modifications to routers.

Overlay networks completely sidestep these two drawbacks. Overlay networks avoid the issue of “dumb” IP routers by performing packet routing and duplication in edge nodes. In these systems, cooperating servers throughout the Internet act as routers in an overlay network.

A reference model for overlay networks, which is capable of modelling all existing approaches in this domain, is presented in chapter 3. A novel framework for topology-aware overlay networks, aiming to maximize path independence without degrading performance is descript in chapter 4. Chapter 5 analyses a framework includes two methods that are used to detect stream inconsistencies in multicast data delivery, and one method that can be used to identify the exact location of the faults. In the following chapters we present a software module, called overlay socket (Chapter 6), the functionality of overlay software (Chapter 8) and the benefits and drawbacks of overlay networks in Chapter 9. The conclusions of the bibliographical research are presented in the last chapter.

2. Overlay Networks

An overlay network is a virtual network of nodes and logical links that is built on top of an existing network with the purpose to implement a network service that is not available in the existing network [VII].

Figure 1-1 demonstrates an overlay network. Just as a physical network has a topology consisting of the nodes of the network and the links between them, an overlay network has a virtual topology, which exists by the agreement of the overlay nodes.

Packets are transmitted only along the virtual links between the overlay nodes using the underlying unicast mechanism provided by IP.

In contrast to the Internet, in which routers are shared resources that cannot be specialized for a particular purpose, the members of an overlay network may provide specialized services specific to the application at hand. An overlay-based multicast system can duplicate packets in the servers, a content distribution network can cache gigabytes of data, RON provides resilient routing by constant performance measurements among participating nodes.

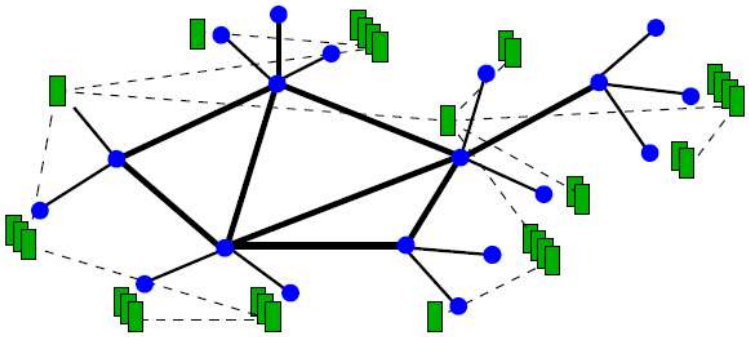


Figure 1: An overlay network. Rectangular end-hosts and dashed links form an overlay network over the physical network of round routers and solid links.

3. Conceptual Model for Overlay Networks

Using an overlay network has the advantage of supporting application specific identifiers and semantic routing, and offers the possibility to provide additional, generic services for supporting network maintenance, authentication, trust, etc., all of which would be very hard to integrate into and support at the networking layer. This reference model for overlay networks is capable of modelling all existing approaches in this domain [I].

In any overlay network a group of peers P provides access to a set of resources R by mapping P and R to an (application-specific) identifier space I using two functions FP : P → I and FR : R→ I. These mappings establish an association of resources to peers using a closeness metric on the identifier space. To enable access from any peer to any resource a logical network is built, i.e., a graph is embedded into the identifier space. These basic concepts of overlay networks are depicted in Figure 2 [X].

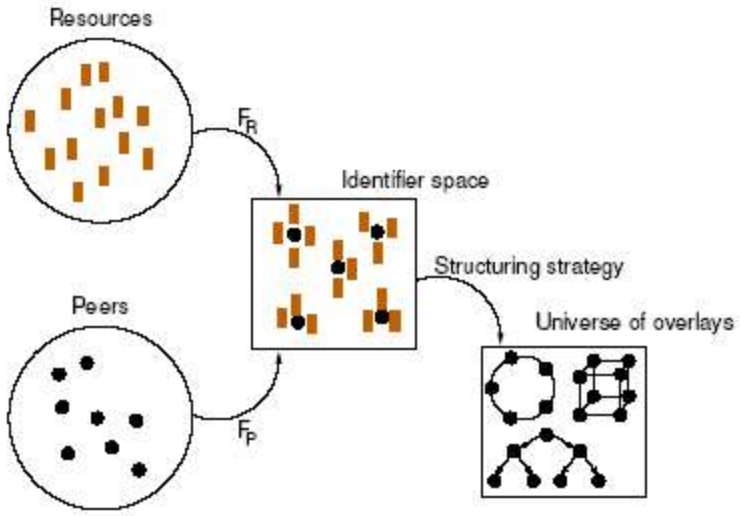


Figure 2: Overlay network design decisions

Each specific overlay network is characterized by the decisions made on the following six key design aspects:

1. Choice of an identifier space
2. Mapping of resources and peers to the identifier space
3. Management of the identifier space by the peers
4. Graph embedding (structure of the logical network)
5. Routing strategy
6. Maintenance strategy

In taking these design decisions the following key requirements for overlay networks are addressed:

Efficiency: Routing should incur a minimum number of overlay hops (with minimum "physical" distance) and the bandwidth (number and size of messages) for constructing and maintaining the overlay should be kept minimal.

Scalability: The concept of scalability includes many aspects. We focus on numerical scalability, i.e., very large numbers of participating peers without significant performance degradation.

Self-organization: The lack of centralized control and frequent changes in the set of participating peers requires a certain degree of self-organization, i.e., in the presence of churn the overlay network should self-reconfigure itself towards stable configurations. This is a stabilization requirement as external intervention typically is not possible.

Fault-tolerance: Participating nodes and network links can fail at any time. Still all resources should be accessible from all peers. This is typically achieved by some form of redundancy. This is also a stabilization requirement for the same reason as above. Fault-tolerance implies that the partial failure property of distributed systems [19] is satisfied, i.e., even if parts of the overlay network cease operation, the overlay network should still provide an acceptable service.

Cooperation: Overlay networks depend on the cooperation of the participants, i.e., they have to trust that the peers they interact with behave properly in respect to routing, exchange of index information, quality of service, etc.

3.1 Choice of Identifier Space

A central decision in designing an overlay network is the selection of the virtual identifier space I. The choice of the virtual identifier space is important for several reasons [I]:

- Addressing: The identifier space plays the role of an address space used for identifying resources in the overlay network. Each peer and resource in an overlay network receives a virtual identifier taken from I (explicitly or implicitly).
- Scalability: To support very large systems, I has to be very large. Through a mapping F_P each peer with a physical address in P is assigned a virtual identifier from I. This is an application of the well-known principle of indirection for achieving numerical scalability.
- Location-independence: The virtual identifier space allows peers to communicate which each other irrespective of their actual physical location. This addresses physical address changes and enables mobility.
- Clustering of resources with peers: The closeness metric d enables the clustering of resources with peers based on proximity.
- Message routing: Virtual identifiers and the closeness metric d are essential for realizing efficient routing.
- Preservation of application semantics: As virtual identifiers can be defined in an application-specific way, application semantics, for example, "proximity" of resources (clustering), can be preserved.

4. Topology - Aware Node Placement

This framework explicitly designs overlay networks to maximize path independence without degrading performance so that it can allow us to better utilize multi-homing at endpoints. To achieve this goal, we measure the diversity between different Internet Service Providers (ISPs) and also between different overlay nodes inside each ISP. Based on these measurements, was developed the topology-aware node placement to ensure path diversity. This allows us to avoid path failures which are not avoidable using currently existing overlay-based approaches [VI].

The effectiveness of overlay networks depends on the natural diversity of overlay paths. However, several recent studies, have observed a high possibility of overlaps among overlay paths when overlay nodes are randomly deployed. Hence, deploying overlay nodes, without consideration of the underlying IP topology, limits the network's ability to recover from path outages and network congestion. One solution to this problem might be to deploy overlay nodes on all routers at all ISPs and dynamically use the overlay nodes as backup paths depending on the source and destination. However, this is impractical due to the deployment overhead and associated economic costs.

In particular, we examine two related questions:

a. Which ISPs and how many ISPs will we deploy the overlay nodes on? Would deploying at more ISPs provide significant gains?

b. For each selected ISP, which and how many routers will we select to deploy overlay nodes at?

4.1. Measurement Methodology

To determine the quality of each overlay node, n_i, we use two metrics: path diversity and latency. For path diversity, we compute the number of shared routers between the direct path and the indirect path through n_i. With the latency metric, the quality of n_i is defined as the round-trip time difference between the direct path and the indirect path via n_i. To gather the direct and indirect path information, we rely on two data sets, D1 and D2, respectively, as described below.

a) Data set D1*:* To measure the direct Internet paths, we collect trace route and ping data from 100 PlanetLab nodes at stub networks. PlanetLab is an open, globally distributed testbed for deploying and accessing planetary-scale network services. We consider these nodes as our target customer networks/endhosts and run traceroute from these points to 1) every other PlanetLab node and 2) top 100Web sites, as shown in Figure 3(a).

b) Data set D2*:* To evaluate the impact of the choice of overlay nodes, we collect another set of traceroute and ping data from topologically and geographically diverse vantage points located in various ISPs. Using the looking glasses, we can access 232 routers from 10 ISPs. We consider these routers as possible places to deploy overlay nodes. Note that routers within the ISP are also geographically distributed. We trigger traceroutes from these points to 1) 100 PlanetLab nodes and 2) top 100 Web sites, as shown in Figure 3(b).

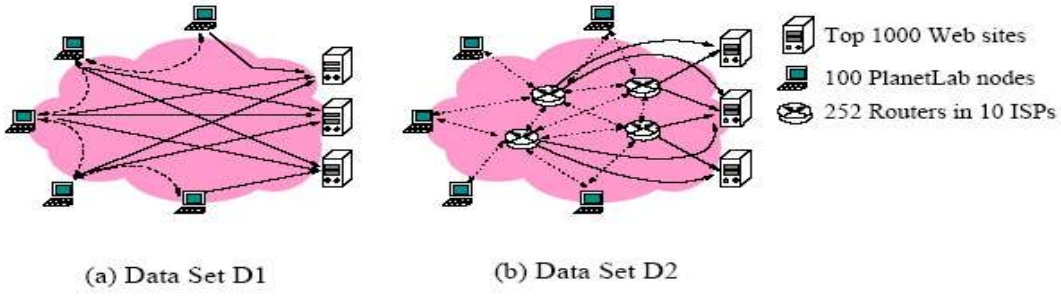


Figure 3: Data sets

4.2 Placement of overlay nodes inside an ISP network

In this section, we attempt to answer the question: which and how many routers should we select from each ISP? We evaluate overlay nodes inside a single ISP with respect to both path diversity and latency.

To measure path diversity, we rely on traceroute data included in the two data sets, D1 and D2.We count the number of overlapping routers between the direct path and the indirect path through a given overlay node as a path diversity metric. In Figure 4, we show the measured path diversity for only 3 ISPs due to space limitations. The x axis indicates the number of shared routers and the y axis represents the cumulative fraction of source and destination pairs. Each line, except the left most line, represents the case where the corresponding overlay node is statically selected for all samples. On the other hand, the left most line represents the optimal case where we intelligently (or dynamically) select the optimal overlay node depending on the source and destination.
From this experiment, we found that:

- Each line in Figure 4 is very close to every other line. This indicates that regardless of which router we select, the overall path diversity provided by the individual overlay node is almost the same.
- On the other hand, the dynamic selection of overlay nodes, represented as the leftmost line in each graph of Figure 4, shows much better path diversity than the static cases. This implies that although overlay nodes are located within the same ISP, the paths taken from these nodes are different. Hence, it is important to have more than one overlay node within the same ISP. This provides flexibility for choosing a proper overlay node.

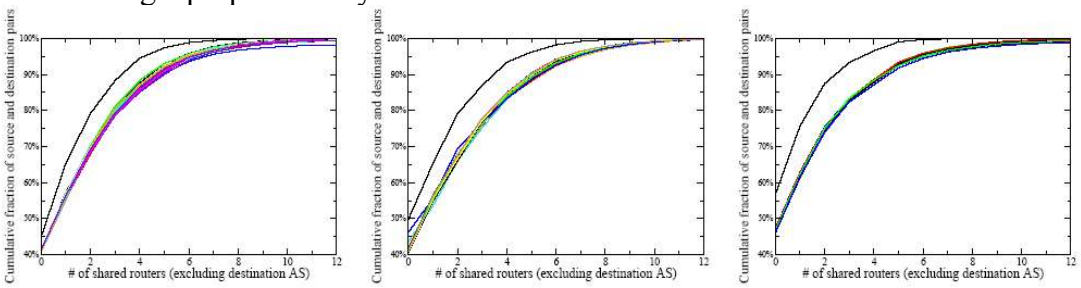


Figure 4: Comparing path diversity of different overlay nodes within the same ISP

5. Fault Types and Causes

One of the primary concerns in using an overlay network is the trustworthiness of its data delivery. Since individual overlay nodes may be owned by anonymous users, data delivery over such a system is potentially exposed to a great number of faults, ranging from innocent data errors to intentional modifications. It is difficult to detect when these faults happen and perhaps even more challenging to repair, because there is no central management or control to oversee the operations in such highly distributed systems [V].

The framework includes two methods that are used to detect stream inconsistencies in multicast data delivery, and one method that can be used to identify the exact location of the faults.

5.1 Fault Detection

The two fault detection methods are based on the following two basic principles used by the majority of fault-tolerant systems:

• **Comparison**: two or more modules that perform the same operation compare their results. If there is disagreement, an error has been detected. Methods such as majority vote can then be used to pinpoint the faulty modules.

• **Self-checking**: a single module can validate the result of its own operation by carrying out an additional

check using redundant information. A simple example of this method is the detection of corrupted information with the use of error detection codes.

5.1.1 Comparison Method

A stream deviation is detected if during the comparison phase there is an indication of a missing packet, a modified packet, or an additional packet. Although the comparison method is conceptually simple, applying it to overlay data delivery imposes a great challenge. The number of nodes in an overlay network can be large, which prohibits packet comparisons among all the nodes. Furthermore, the amount of data to be compared can be immense.

a. Stream Comparison with Bloom Filters A naive solution of packet-to-packet comparison may increase the overall network traffic at least *t* times, where *t* is the number of comparisons required for each packet.

b. Node Selection The detection power of the comparison method depends on the nodes selected for the Bloom filters exchange. The most suitable nodes are the data stream sources, but that solution does not scale neither with the number of sources nor with the total number of overlay node

5.1.2 Self-Checking Method

The intuition behind the self-checking method is to transmit some additional information related with one packet, which can be the checksums or a hash value of the packet payload, and to verify the integrity of the packet based on that information.

5.2 Fault Repair

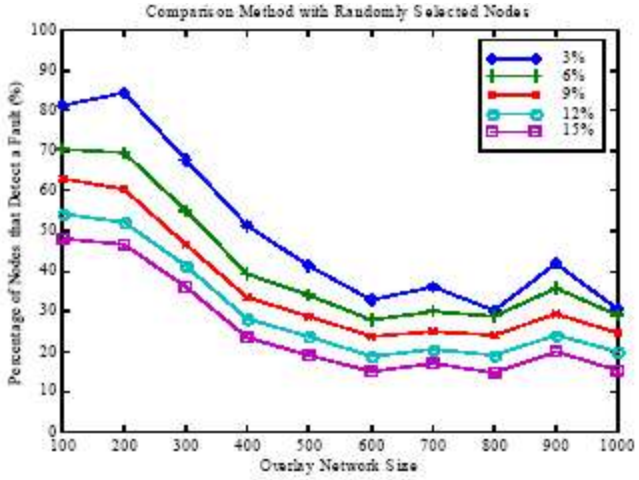
Whenever a faulty or malicious node is detected the non faulty nodes need to isolate the problematic node, by removing it out of any forwarding path. Thus, fault repair is a two step process: first, when a node detects a deviation in the data stream, it initiates a procedure to pinpoint the faulty node, and second, in collaboration with the other nodes puts the faulty node aside. The identification of the faulty node is done by querying a number of upstream nodes along the path of the deviated stream, which are n = 2k(k = 1, 2, 3...) hops away. The query takes the form of a request for the Bloom filter for the recently received packets. With those filters a node is able to check which nodes received the same faulty packets, and thus to estimate its distance to the faulty node. That distance is expressed as number of overlay hops. In the case that the comparison method is used for the fault detection, then the querying step can be omitted, given that the Bloom filters are already known. On the other hand, the self-checking method can detect the faults almost instantly.

5.3 Simulation

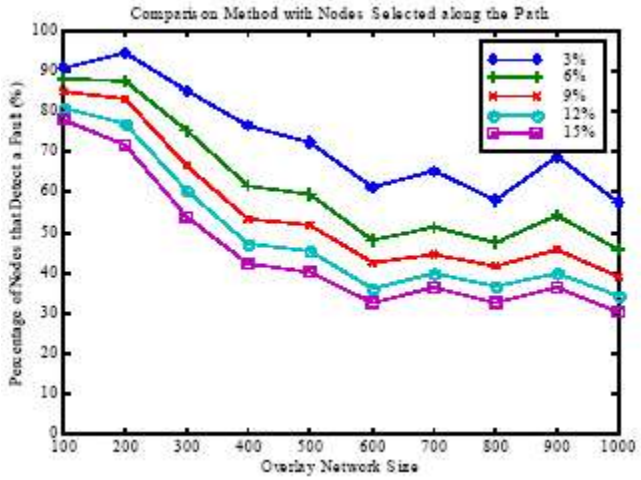
We use two metrics to measure each method's fault detection power. The first metric is the percentage of nodes that can detect a fault in a given network with some malicious nodes. The other is the rate of false positives. Three methods are compared in the simulation: the comparison method with randomly selected nodes, the comparison method, in which nodes to be inquired are on the path to the source, and the self checking method [V].

In order to get the lower bound of methods' performance, two strong assumptions were made on the attack model. First, we assume all malicious nodes are in full collusion. Each malicious node modifies not only data payload, but also the checking information

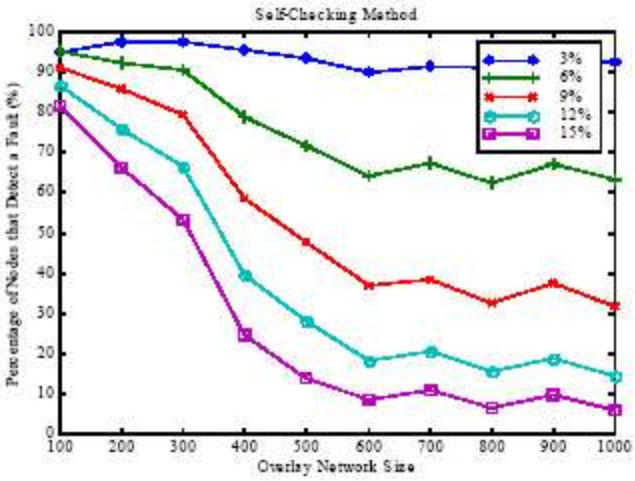
carried by subsequent packets to match the modified payload. Different malicious nodes will make consistent modifications, so that checking information forwarded by one matches the data payload forwarded by another. Second, we assume the number of malicious nodes increases linearly with the network size, that is, their percentage keeps unchanged as the network grows. In reality, it is more likely that a large overlay network has a very small number of malicious nodes, of which only a few collude with each other.



(a) Comparison Method: Randomly selected nodes



(b) Comparison Method: Selected nodes belong on the path to the root



(c) Self-Checking Method

Figure 5: Percentage of nodes that can detect a fault in a network of 11 disjoint paths

6. Overlay Socket

Application-layer overlay networks provide flexible platforms for developing new network services without requiring changes to the network layer infrastructure. Members of an overlay network, which can be hosts, routers, servers, or applications, organize themselves to form a logical network topology, and communicate only with their respective neighbours in the overlay topology. A member of an overlay network sends and receives application data, and also forwards data intended for other members [XII].

The diversity and complexity of building and maintaining overlay networks make it impractical to assume that application developers can be concerned with the complexity of managing the participation of an application in a specific overlay network topology.
The software module, called *overlay socket*, intends to simplify the task of overlay network programming. The design of the overlay socket pursues the following set of objectives: First, the application programming interface (API) of the overlay socket does not require that an application programmer has knowledge of the overlay network topology. Second, the overlay socket is designed to accommodate different overlay network topologies. Switching to different overlay network topologies is done by modifying parameters in a configuration file. Third, the overlay socket, which operates at the application-layer, can accommodate different types of transport layer protocols. This is accomplished by using *network adapters* that interface to the underlying transport layer network and perform encapsulation and de-encapsulation of messages exchanged by the overlay socket. Currently available network adapters are TCP, UDP, and UDP multicast. Fourth, the overlay socket provides mechanisms for bootstrapping new overlay networks.

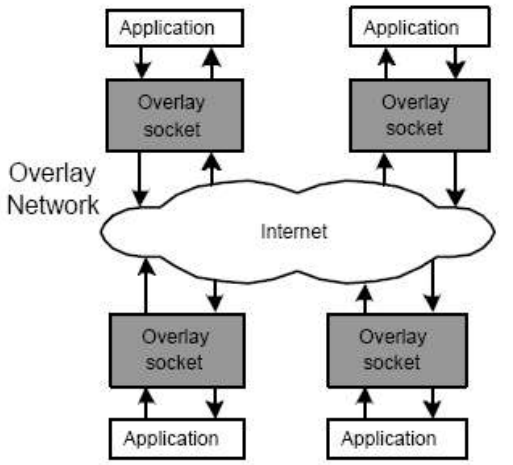


Figure 6: The overlay network is a collection of overlay sockets.

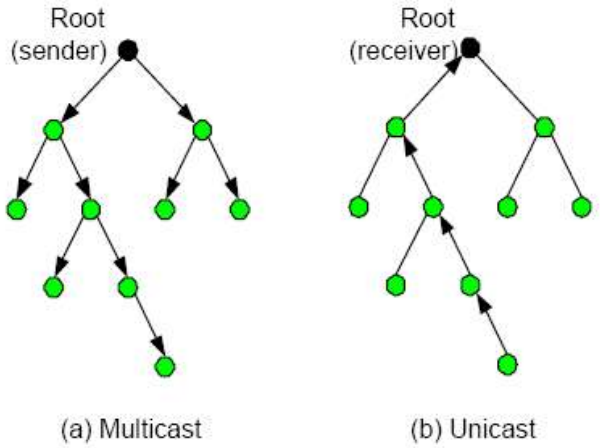


Figure 7: Data forwarding in overlay networks.

6.1 Basic Concepts

An overlay socket is an endpoint for communication in an overlay network, and an overlay network is seen as a collection of overlay sockets that self-organize using an overlay protocol (see Figure 6). Each overlay socket executes an overlay protocol that is responsible for maintaining the membership of the socket in the overlay network topology. Each overlay socket has a logical address and a physical address in the overlay network. The logical address is dependent on the type of overlay protocol used. The physical address is a transport layer address where overlay sockets receive messages from the overlay network. When an overlay socket is created, the socket is configured with a set of configuration parameters, called attributes. The configuration file specifies the type of overlay protocol and the type of transport protocol to be used, but also more detailed information such as the size of internal buffers, and the value of protocol-specific timers. The most important attribute is the overlay identifier (overlay ID) which is used as a global identifier for an overlay network and which can be used as a key to access the other attributes of the overlay network.

Overlay sockets exchange two types of messages, protocol messages and application messages. Protocol messages are the messages of the overlay protocol that maintain the overlay topology. Application messages contain application-data that is encapsulated in an overlay message header. If an overlay socket receives an application message from one of its neighbours in the overlay network, it determines if the message must be forwarded to other overlay sockets, and if the message needs to be passed to the local application. The transmission modes currently supported by the overlay sockets are unicast, and multicast. In multicast, all members in the overlay network are receivers. For example, a multicast message is transmitted downstream a spanning tree that has the sender of the multicast message as the root (see Figure 7(a)). When an overlay socket receives a multicast message, it forwards the message to all of its downstream neighbours (children) in the tree, and passes the message to the local application program. A unicast message is transmitted upstream a tree with the receiver of the message as the root (see Figure 7(b)). An overlay socket that receives a unicast message forwards the message to the upstream neighbour (parent) in the tree that has the destination as the root.

7. Global Flow Control for Wide Area Overlay Networks

The Cost-Benefit framework is a new global flow control strategy for wide area overlay network multicast based on "economic" principles and competitive analysis. This framework assigns costs to network resources, and benefits to achieving user goals such as multicasting a message to a group or receiving a message from a group. Intuitively, the cost of network resources, such as buffers in routers, should go up as the resource is depleted. When the resource is not utilized at all, its cost should be zero. When the resource is fully utilized, its cost should be prohibitively expensive. Finding the best cost function is an open question [2].

The algorithm decides to allow use of resources if the benefit attached to that use is greater than the total cost of allowing the use. The choice of benefit function allows us to optimize for various goals. By adjusting the benefit function, performance issues such as throughput, as well as policy issues such as fairness, can be taken into account when making flow control decisions. In this way the framework allows adjustable trade-offs between conflicting properties such as fairness and throughput. The simulations use the ns2 simulator and examine the behaviour of several overlay network configurations

The overlay network model used is a graph with nodes and overlay links. Each node on the graph represents a host running a daemon program. Each overlay link is a unicast link between two nodes, which may be a long path traversing multiple routers and physical links in the Internet as is seen in Figure 8. Each daemon chooses a tree from this graph based on the network topology, in which it will multicast messages.

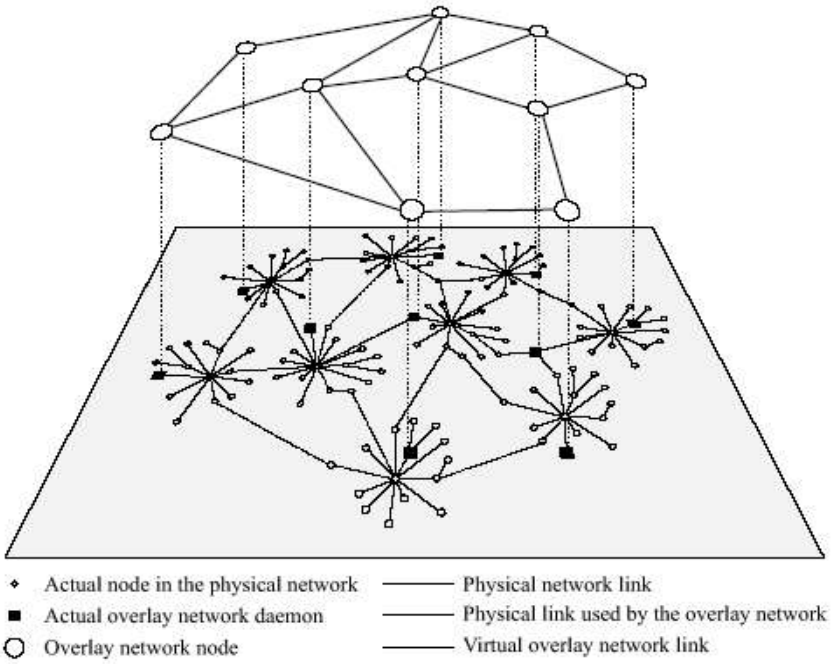


Figure 8: Overlay Network Architecture

The algorithmic foundation can be summarized as follows: We price links based on their "opportunity cost", which increases exponentially with link utilization. We

compare different connections based on the total opportunity cost of the links they use, and slow down connections with large costs, by delaying their packets at the entry point.

We chose buffer space in each overlay network router as the scarce resource we want to control. Conceptually, we model our software overlay network router as a router with fixed size output link buffers where packets are placed into the appropriate output link queues as soon as the packet is received by the router.

Each router establishes the cost for each of its outgoing links and advertises this cost to all the other daemons using the overlay network. The price for a link is zero if its corresponding buffer is empty. This means that the cost is zero as long as the link is not congested, i.e. the link can accommodate all the incoming traffic. As the link gets congested and messages accumulate in the buffer, the cost of the link increases. The price can theoretically go as high as infinite when the buffer is full. In practice, the cost of the link will increase until a given value when nobody will be able to buy it.

The utilization of a link is given by where is the average number of messages in the buffer, is the desired capacity of the buffer. In the implementation the minimum benefit is. So the cost is:

$$C(x) = \alpha \cdot \gamma^{x/M} = \beta^{x/M}$$

This function ranges from 1 to β. We want to scale it to range from 0 to S (the prohibitive cost). So the cost becomes:

$$C(x) = S \cdot \frac{\beta^{x/M} - 1}{\beta - 1}$$

Using a small base for the exponent has real-world advantages and still fits the model. First, it provides an increase in costs almost immediately as the link first becomes congested. Second, it is still exponential, which is required by the theoretical model. We chose as the base of the exponent. So finally we get:

$$C_l(x) = S \cdot \frac{e^{x/M} - 1}{e - 1}$$

8. Functionality of overlay software

Overlay networks form a layer for organisation and communication in distributed applications. This section describes their different levels of functionality as illustrated in figure 9. While the development process of overlay software deals with all of them, only few level are well supported by design aids and frameworks. The lowest two levels comprise the general operating system support for Internet-level network I/O and edge-level message passing. These levels are not specific to overlays and are usually hidden by higher layers. A number of overlays, such as Bamboo, are implemented on top of generic event-driven state machines like SEDA that model message processing in Internet servers. While EDSMs were not designed for overlay development, they still provide a good abstraction level for scalable event processing [XIII].

Overlay routing protocols then deal with local routing decisions for scalable end-to-end message forwarding. They are distributed algorithms, executed at each member node, with the purpose of forwarding messages at the overlay level from senders to receivers. Routing is left out of figure 9 for clarity reasons.

Functionality	Support in current frameworks	
Topology Selection	Node views	
Topology Adaptation		
Topology Maintenance		(Macedon)
Topology Rules		
Message Processing	iOverlay, Macedon	Flow-Graphs, EDSMs, ...
Message Passing		Serialisation, RPC, CORBA, ...
Network I/O		Sockets, TCP/UDP, ...

Figure 9: Framework Support for Overlay Software

8.1 Node Views, the System Model

The model, Model-View-Controller pattern, is an active local database on each node, a central storage place for all data that a node knows about remote nodes. Once the data is stored in a single place, software components no longer have to care about any data management themselves. They benefit from a locally consistent data store and from notifications about changes. The major characteristics of the overlay topology are then defined in views of the database. They represent sets of nodes that are of interest to the local node (such as its neighbours). Different views provide different ways of selecting and categorising nodes, and different ways of adapting topologies. Topology selection is then mainly a matter of selecting the right set of views.

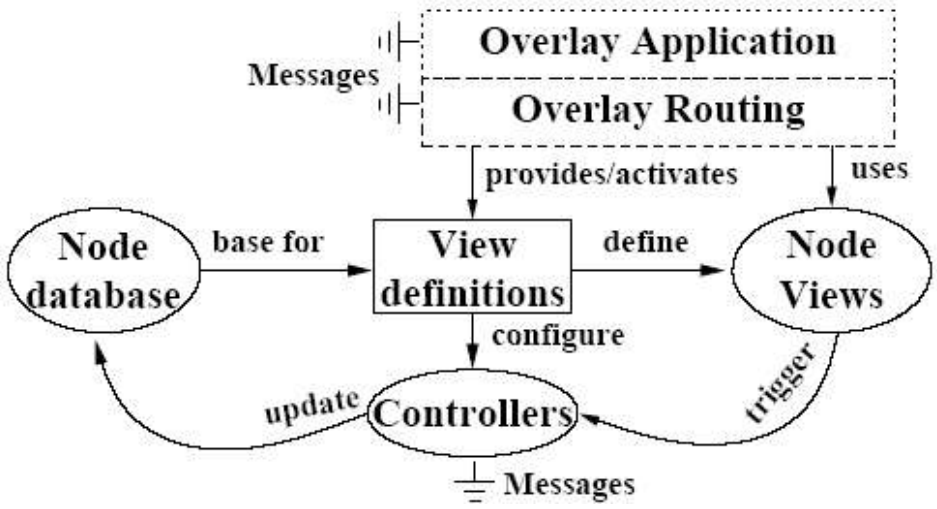


Figure 10: Components of the system model

As the views form the most important overlay specific part of the implementation, they are also the most crucial part for an abstract and framework independent specification. Their definition is the main goal of the Slosl language that is briefly presented in the next section.

8.2 SLOSL, the View Specification Language

For the view definitions that implement topology rules and adaptation, there is Slosl, the SQL-Like Overlay Specification Language. Here is a simple example, an implementation of the Chord graph. The statements CREATE VIEW, SELECT, FROM and WHERE behave as in SQL. The WHERE clause specifically implements node selection based on node attributes. Note that Slosl is not concerned with the source of the information that node attributes contain. It only constrains and categorises the presentation of locally available data.

```
1  CREATE VIEW chord_fingertable
2  AS SELECT node.id, node.ring_dist, bdist=node.ring_dist−2^i
3  RANKED lowest(backups+i, node.msec_latency / node.ring_dist)
4  FROM node_db
5  WITH log_k = log(|𝒳|), backups = 1
6  WHERE node.supports_chord = true AND node.alive = true
7  HAVING node.ring_dist in (2^i : 2^(i+1))
8  FOREACH i IN (0:log_k)
```

As a major step towards simplified, abstract overlay development, a graphical editor has been designed (fig. 11) based on our system model. It allows the framework independent specification of overlay systems and outputs abstract overlay specifications in OverML, a new XML specification language for node attributes, Slosl statements, messages and EDSM flow descriptions.

Figure 11: The SLOSL Overlay Workbench

9. Benefits and Drawbacks

An overlay network provides advantages over both centrally located solutions and systems that advocate running code in every router [1]. An overlay network is:

I. **Incrementally Deployable**: An overlay network requires no changes to the existing Internet infrastructure, only additional servers. As nodes are added to an overlay network, it becomes possible to control the paths of data in the substrate network with ever greater precision.
II. **Adaptable:** Although an overlay network abstraction constrains packets to flow over a constrained set of links is constantly being optimized over metrics that matter to the application. For instance, the overlay nodes may optimize latency at the expense of bandwidth. The detour project has discovered that there are often routes between two nodes with less latency than the routes offered by today's IP infrastructure Overlay networks can find and take advantage of such routes.
III. **Robust:** By virtue of the increased control and the adaptable nature of overlay networks, an overlay network can be more robust than the substrate fabric. For instance, with a sufficient number of nodes deployed, an overlay network may be able to guarantee that it is able to route between any two nodes in two independent ways. While a robust substrate network can be expected to repair faults eventually, such an overlay network might be able to route around faults immediately.
IV. **Customizable:** Overlay nodes may be multipurpose computers, easily outfitted with whatever equipment makes sense. For example, Overcast makes extensive use of disk space. This allows Overcast to provide bandwidth savings even when content is not consumed simultaneously in different pars of the network.
V. **Standard:** An overlay network can be built on the least common denominator network services of the substrate network. This ensures that overlay traffic will be treated s well as any other. For example, Overcast uses TCP (in particular, HTTP over port 80) for reliable transport. TCP is simple, well understood, network friendly, and standard. Alternatives, such as a "home grown" UDP protocol with retransmissions, are less attractive by all these measures. For better or for worse, creativity in reliable transport is a losing battle on the internet today.

On the other hand, building an overlay network faces a number of interesting challenges. An overlay network must address:

I. **Management complexity:** The manager of an overlay network is physically far removed from the machines being managed. Routine maintenance must either be unnecessary or possible from far, sing tools that do not scale in complexity with the size of the network. Physical maintenance must be minimized and be possible by untrained personnel.
II. **The real world:** In the real world, IP does not provide universal connectivity. A large portion of the internet lies behind firewalls. A significant and growing share of hosts are behind Network Address Translators (NATs), and proxies. Dealing with these practical issues is tedious, but crucial to adoption.
III. **Inefficiency:** An overlay can not be as efficient as a code running in every router. However, our observation is that when an overlay network is small, the inefficiency, measured in absolute terms, will be small as well – and as the overlay networks grows, its efficiency can approach the efficiency of router based services.
IV. **Information loss:** Because the overlay network is built on top of a network infrastructure (IP) that offers nearly complete connectivity (limited only by firewalls, NATs and proxies), we expend considerable effort deducing the topology of the substrate network.

10. Conclusions

Overlay networks are an important way for applications to obtain network behaviour that would otherwise require widespread router modifications. By their very nature, it is possible to deploy overlay networks with no additional support. Yet doing so creates inefficiencies. Path painting and packet reflection address those inefficiencies with simple router extensions that can be used in creative ways to perform packet routing and duplication at appropriate locations in the network.

A number of interesting questions remain to be answered, some of which were implied by the difficulties described in the previous section.

Two ideas in merit further consideration to determine whether their value exceeds their complexity. First, a node may wish to exert some control over the rules that routers use to propagate its reflection request. More generally, a node might wish to scope its reflection requests, providing some minimum and maximum distance for them to be propagated. This feature could be used to ensure that a reflection request propagates exactly to the router that responded to the node's paint request thereby reducing the uncertainty involved in the current scheme.

Second, both primitives might benefit from a mechanism which would allow a node to ask for requests to be sent to it rather than from it. The basic idea behind this notion is to eliminate difficulties due to asymmetric routes by causing requests (reflection and paint) to follow the same paths as the packets they are intended to affect.

Bibliography

[1] J. Jannotti: Network Layer Support for Overlay Networks, August 2002

[2] Y. Amir, B. Awerbuch, C. Danilov, J. Stanton: Global Flow Control for Wide Area Overlay Networks: A Cost-Benefit Approach, August 2001

[3] Z. Li, P Moharata: QRON: QoS-Aware Routing in Overlay Networks, *IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, VOL. 22, NO. 1*, January 2004

[4] M. Castro, P. Druschel, A. Ganesh, A. Rowstron and D. S. Wallach: Secure routing for structured peer-to-peer overlay networks, December 2002

[5] A. Nakao, L. Peterson and A. Bavier: A Routing Underlay for Overlay Networks, August 2003

Internet

I. **http://dks.sics.se/pub/refarch.pdf**

In this paper presents a reference model for overlay networks which is capable of modeling different approaches in this domain in a generic manner. It is intended to allow researchers and users to assess the properties of concrete systems, to establish a common vocabulary for scientific discussion, to facilitate the qualitative comparison of the systems, and to serve as the basis for defining a standardized API to make overlay networks interoperable.

II. **http://www.cs.huji.ac.il/~ittaia/papers/aaabmp.pdf**

This paper presents a generic scheme for a central, yet untackled issue in overlay dynamic networks: maintaining stability over long life and against malicious adversaries. The generic scheme maintains desirable properties of the underlying structure including low diameter, and efficient routing mechanism, as well as balanced node dispersal. These desired properties are maintained in a decentralized manner without resorting to global updates or periodic stabilization protocols even against an adaptive adversary that controls the arrival and departure of nodes.

III. **www.cs.rice.edu/Conferences/IPTPS02/169.pdf**

The environment assumed in this paper is that of a set of cooperating, widely separated peers, running as user-level processes on ordinary PC's or workstations. Peers "export" data, and keys are "mapped" onto overlay servers. The set of peer nodes that export data is also the set of overlay servers. A "node" is a process participating in the system.

IV. **www-db.stanford.edu/~crespo/publications/op2p.pdf**

In a peer-to-peer (P2P) system, nodes typically connect to a small set of random nodes (their neighbors), and queries are propagated along these connections. Such query flooding tends to be very expensive. This paper proposes that node connections be influenced by content, so that for example, nodes having many "Jazz" files will connect to other similar nodes. Thus, semantically related nodes form a Semantic Overlay Network (SON). Queries are routed to the appropriate SONs, increasing the chances that matching files will be found quickly, and reducing the search load on nodes that have unrelated content.

V. **www.cs.ucla.edu/~vpappas/p/vp_icdcs04.pdf**

This paper describes a set of mechanisms designed to detect and repair errors in the data stream. Utilizing the highly redundant connectivity in overlay networks, our design splits each data stream to multiple sub streams which are delivered over disjoint paths. Each sub-stream carries additional information that enables receivers to detect damaged or lost packets. Furthermore, each node can verify the validity of data by periodically exchanging Bloom filters, the digests of recently received packets, with other nodes in the overlay.

VI. **www.eecs.umich.edu/~farnam/pubs/2005-hwj-infocom.pdf**

This paper proposes a novel framework for topology aware overlay networks. In this framework, we expressly design overlay networks, aiming to maximize path independence without degrading performance. It presents measurement-based heuristics for 1) placement of overlay nodes inside an ISP and 2) selection of a set of ISPs. We base our analysis on extensive data collection from 232 points in 10 ISPs, and 100 PlanetLab nodes.

VII. **www.cs.virginia.edu/~cs757/slidespdf/757-09-overlay.pdf**

A presentation of overlay networks, which includes the applications, the benefits and costs. It is also presented a series of case studies.

VIII. **http://dks.sics.se/pub/refarch.pdf**

This paper presents a reference model for overlay networks which is capable of modeling different approaches in this domain in a generic manner. It is intended to allow researchers and users to assess the properties of concrete systems, to establish a common vocabulary for scientific discussion, to facilitate the qualitative comparison of the systems, and to serve as the basis for defining a standardized API to make overlay networks interoperable.

IX. **http://www.cnds.jhu.edu/pub/papers/voip.pdf**

This paper describes two algorithms to improve the performance of such VoIP applications. These mechanisms are used for localized packet loss recovery and rapid rerouting in the event of network failures. The algorithms are deployed on the routers of an application-level overlay network and require no changes to the underlying infrastructure. Initial experimental results indicate that these two approaches can be composed to yield voice quality on par with the PSTN.

X. **http://www.cs.uml.edu/~buford/irtf-p2prg/p2psip/JBuford-IETF-P2PSIP-Overlay-Systems-v3.pdf**

A presentation of overlay networks, of overlay design space and unstructured overlay networks.

XI. **http://www.cs.princeton.edu/courses/archive/fall03/cs597B/handouts/pdn03-012.pdf**

This paper proposes a set of primitive operations and three library routing services that can be built on top of them, and describes how such libraries could be useful to overlay services.

XII. **http://www.cs.virginia.edu/~mngroup/pub/ngc_2003.pdf**

This paper presents the concept of an *overlay socket* as a new programming abstraction that serves as the end point of communication in an overlay network. The overlay socket provides a socket-based API that is independent of the chosen overlay topology, and can be configured to work for different overlay topologies. The overlay socket can support application data transfer over TCP, UDP, or other transport protocols. This paper describes the design of the overlay socket and discusses API and configuration options.

XIII. **http://www.dvs1.informatik.tu-darmstadt.de/publications/pdf/behnel2005overlayspecification.pdf**

This paper presents a novel approach to overlay implementation by modeling topologies as a distributed database. This approach, named “Node Views”, abstracts from low-level issues like I/O and message handling. Instead, it moves ranking nodes and selecting neighbors into the heart of the overlay software development process. It decouples maintenance components in overlay

software and allows implementing them in a generic, configurable way for pluggable integration in frameworks.

XIV. **http://www.cse.psu.edu/~wlee/Publications/wlee%20ICNP04.pdf**

XV. **http://nms.lcs.mit.edu/papers/ron-sosp2001.pdf**

A Resilient Overlay Network (RON) is an architecture that allows distributed Internet applications to detect and recover from path outages and periods of degraded performance within several seconds, improving over today's wide-area routing protocols that take at least several minutes to recover. A RON is an application-layer overlay on top of the existing Internet routing substrate. The RON nodes monitor the functioning and quality of the Internet paths among themselves, and use this information to decide whether to route packets directly over the Internet or by way of other RON nodes, optimizing application-specific routing metrics.

XVI. **http://research.microsoft.com/~antr/MS/eclipse.pdf**

This paper discusses the impact of the Eclipse attack on several types of overlay and it proposes a novel defense that prevents the attack by bounding the degree of overlay nodes. This defense can be applied to any overlay and it enables secure implementations of overlay optimizations that choose neighbors according to metrics like proximity. It presents preliminary results that demonstrate the importance of defending against the Eclipse attack and show that this defense is effective

XVII. **http://www.cs.utah.edu/classes/cs6935/papers/underlay.pdf**

This paper proposes a set of primitive operations and three library routing services that can be built on top of them, and describes how such libraries could be useful to overlay services.

XVIII. **http://www.cs.rice.edu/Conferences/IPTPS02/151.pdf**

This paper presents the initial architecture of a brocade secondary overlay on top of a Tapestry network, and demonstrates its potential performance benefits by simulation.

XIX. **http://project-iris.net/irisbib/papers/failuredetection:infocom05/paper.pdf**

This paper studies how the design of various keep-alive approaches affect their performance in node failure detection time, probability of false positive, control overhead, and packet loss rate via analysis, simulation, and implementation. In this paper is found that among the class of keep-alive algorithms that share information, the maintenance of back pointer state substantially improves detection time and packet loss rate.

XX. **http://dks.sics.se/pub/mcast.pdf**