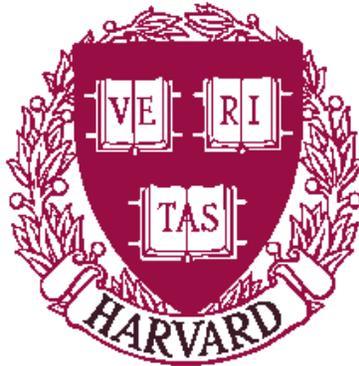


**”Blossom: A Decentralized Approach to
Overcoming Systemic Internet
Fragmentation”**

Geoffrey Goodell
Scott Bradner
Mema Roussopoulos

TR-10-05



Computer Science Group
Harvard University
Cambridge, Massachusetts

Blossom: A Decentralized Approach to Overcoming Systemic Internet Fragmentation

Geoffrey Goodell
goodell@eecs.harvard.edu
Harvard University, Cambridge, MA

Scott Bradner
sob@harvard.edu
Harvard University, Cambridge, MA

Mema Roussopoulos
mema@eecs.harvard.edu
Harvard University, Cambridge, MA

Abstract

The Internet is systemically fragmented. We consider the causes of fragmentation, including both technical concerns, such as middleboxes and routing failure, as well as political concerns, such as incomplete peering and the structure of Internet governance. While fragmentation may be desirable in certain circumstances and for various reasons, it can also be problematic, violating central Internet design principles and rendering routine tasks difficult. We motivate the need for a system designed to facilitate connectivity throughout the Internet, providing the benefits of locality, universal access, and distributed management, while interoperating with the existing infrastructure. Finally, we describe our prototype implementation that enables overcoming fragmentation in our network testbed.

I. INTRODUCTION

One of the central design principles of the Internet is that access to resources should be a function of *who* users are rather than *where* users are, and to enable access, each resource should have a globally unique identifier. The modern Internet does not allow for such an arrangement. Instead, the set of resources to which a given user has access is a function of the location of the user's host within the network topology. That is, the network itself acts to restrict access to certain resources to hosts in particular locations. In this sense, we say that the Internet is *fragmented*.

There are many causes of fragmentation, ranging from accidental (routing failures, misconfigured policies, unreliable network elements) to deliberate (content filtering, network address translation, firewalls, malicious service providers). We assert that fragmentation is inevitable and describe how prevailing Internet architecture fails to avoid fragmentation despite various efforts to provide consistency.

In this paper, we propose Blossom, a peer-to-peer overlay network of *forwarders* that carry TCP traffic. Blossom forwarders act as intermediaries between nodes that cannot communicate directly, forming an unstructured overlay to provide peer-to-peer communication across middleboxes. We describe the design goals, architecture, and a prototype implementation of Blossom that leverages the Tor network, a collection of about 150 hosts spread across fifteen countries, as a testbed. We show how Blossom can be used to defeat both perfunctory and accidental fragmentation without requiring changes to client or server applications, and we show how it is possible

to use the same system to present the Internet from different points of view.

In Blossom, not all participants have the same idea of the set of resources provided by the Internet, what region of the Internet constitutes the “core”, or which set of real-world organizations are responsible for Internet governance. We seek to promote the idea of an Internet whose management reflects the management of the physical world rather than imposing organizational structure where such structure need not exist. We also seek to provide a means by which the names used to identify resources in one area of the Internet do not unnecessarily restrict the names used to identify resources in other areas.

One of our primary goals is to make it easier for providers to design reasonable policies that are less tightly integrated with routing mechanism. We achieve this goal by providing an infrastructure that allows connections between nodes using locally meaningful identifiers. We envision a coreless Internet that does not draw a distinction between “internal” and “external” networks. We thus wish to devise a useful tool for allowing individual internet peer nodes to provide policy-compliant access to services without requiring special configuration of the network infrastructure. One might argue that this technology can be used to provide open access to networks protected by firewalls, but the reality is that technology that can create tunnels through firewalls exists today. What our system provides is a means by which otherwise-inaccessible resources can be accessed in a general way, without preordained arrangement between nodes on both sides of the obstacle.

Ultimately, Blossom should enable a transition to better policy: because the Blossom forwarders are not part of the underlying network-layer routing infrastructure, they can be deployed in a manner that allows policy settings that are both more flexible and closer to the endpoints. For example, an organization currently using a firewall to block all incoming traffic to protect a small set of resources behind its firewall could configure a Blossom forwarder that provides more fine-grained access to resources on a resource-by-resource basis, making both “opt-in” and “opt-out” network policies easier to implement. Additionally, an organization could encourage the use of Blossom forwarders by its various departments in order to implement a multi-tiered security framework that does not rely upon central administration for every policy decision.

II. BACKGROUND

A. Causes of Fragmentation

Systemic Internet fragmentation has many causes, including but not limited to the following:

Interdomain routing misconfiguration. Misconfiguration of routers that participate in the Border Gateway Protocol [18], [21] is a significant cause of observed routing failures including unreachable and suboptimal routes [17], [13], [15], [11].

Interdomain routing instability. Interactions between BGP policies often lead to persistent interdomain routing oscillation [23], [12] and interdomain routing oscillation in turn leads to degraded network performance [13].

Interdomain routing policy. Accidental misconfiguration of interdomain routing policies, as well as purposeful configuration of these policies due to financial or political reasons, can lead to an incomplete set of available routes.

Firewalls. Firewalls deployed at routers block incoming traffic from entering the network behind the router. A firewall can thwart attacks that involve random automated probes for services with vulnerabilities over an address range, but this comes at the cost of effectively enacting a policy, even when such policy is not required by organizational goals.

Network Address Translation. Organizations often deploy NAT for the same “security reasons” used to justify the deployment of firewalls. NATs violate the end-to-end principle [19], translating endpoint identifiers so that peer nodes do not know with whom they are communicating and making systems at the ends of the network dependent upon the reliability of systems in the interior of the network.

Content Filtering. Some firewalls and other devices are configured to filter packets based upon application-layer content (for example, filtering HTTP requests for particular URLs); this technique can be used for large-scale censorship of sensitive content [3]. The scale at which governments and providers are considering deployment of such technologies indicates the potential for misuse.

Explicit Address Filtering. Internet service providers may also configure their routers to explicitly block packets based on source or destination addresses. Deploying such filtering technology in the middle of the network suggests greater distance between those subject to the policy and those enacting it.

Transparent Proxies and Caches. Transparent web proxies often cache replies to improve performance, but may fail to issue cache-control directives to specify that a user does not want a cached copy [1]. Moreover, proxies pose a threat to network transparency by injecting intelligence into the center of the network [2].

Anycast. In anycast, the network chooses to which server to forward a request on behalf of a client. While this might have certain efficiency benefits, the client is unable to specify the exact server it desires to use, and the end-to-end principle suggests that this should be possible [19]. Also, peers have no means of determining to which server it is conversing in case of problems.

DNS hacks. DNS can be used to provide different IP addresses for the same hostname based upon the location of the requester in the network. This makes it hard for the requester to discover and access a resource at an IP address that is in a remote location.

B. Related Work

Blossom takes a unique approach to addressing a set of well-established problems:

- **INDIRECTION.** In I3 [22], services are registered with the infrastructure. TRIAD [4] uses globally unique, hierarchical names to identify networks; these names are propagated throughout the system via BGP-like advertisements among TRIAD nodes. Blossom does not require registration of services, names of resources need not be globally unique, and names of Blossom forwarders are non-hierarchical.
- **ANTI-CENSORSHIP.** Infranet [8] and Tor [7] aim to anonymize communication. While anonymity is not a direct goal of Blossom, anti-censorship systems can help prevent intermediaries from fragmenting the network.
- **DECOUPLING POLICY FROM MECHANISM.** FARA [5] provides a general framework for describing associations between nodes without requiring a global namespace. Platypus [20] provides a system for enforcing routing policy on the forwarding plane rather than the control plane, relying upon cooperation from intermediary ISPs. Blossom aims to not require such cooperation, at least not on a technical level.
- **INTEROPERATING WITH MIDDLEBOXES.** UIP [9] and DOA [25] aim to route around middleboxes, providing efficiency and scalability in the process. Unlike Blossom, these systems require modification to the protocol stack.
- **NON-UNIVERSAL NAMESPACES.** Semantic-Free Referencing [24] stipulates that resources have globally-unique self-certifying names that can be resolved by clients into semantic-free tags using third-party services that are not universal. The goal is to decouple the name of a resource from its content; note that this is subtly different from the *locality* goal of Blossom.

III. DESIGN GOALS

We submit that the amorphous nature of the Internet facilitates its growth, that fragmentation is part of this amorphous nature, and that designing an architecture that acknowledges fragmentation as a fundamental characteristic of the underlying network provides a number of benefits. We outline these below.

Locality. *Multiple services with the same name may coexist within different local namespaces.* In the physical world, the meaning of a name is dependent upon its context. However, the existing Internet paradigm intends for there to exist a global namespace in which centralized authorities allocate names hierarchically and uniquely. We believe that a system that facilitates communication across network fragments should also allow for the development of distinct local namespaces,

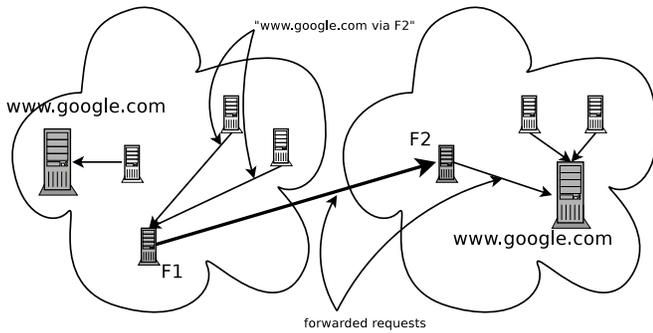


Fig. 1. LOCALITY. Multiple services with the same name may coexist within different local namespaces. (Meaningful names within a local space.)

in which names have local meaning, while also allowing access to objects in other namespaces that happen to bear the same name. For example, in Figure 1, there are two resources named `www.google.com` in the left and right fragments. The service provided by each resource should not be required to be the same. Instead, a host in the left fragment should be able to access the `www.google.com` resource in the right fragment via the Blossom forwarder $F2$. This may afford businesses the opportunity to protect their trademarks, avert some Internet namespace arbitrage, and generally lead to relaxation of an unnatural constraint on naming.

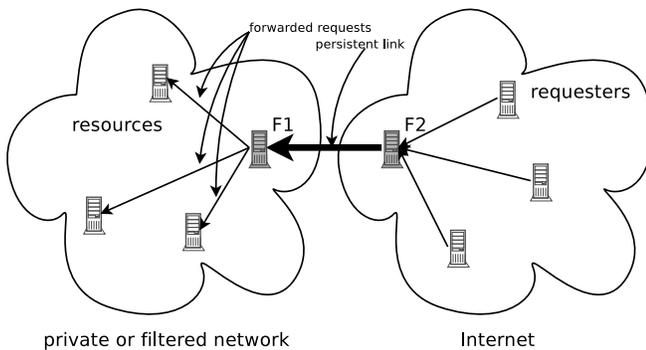


Fig. 2. UNIVERSAL ACCESS. If two hosts can both access forwarders within the same forwarding infrastructure, then those two hosts can use the infrastructure to communicate. (Circumvent technical barriers.)

Universal Access. If two hosts can both access forwarders within the same forwarding infrastructure, then those two hosts can use the infrastructure to communicate. Sometimes, communication between networks is compromised for architectural convenience rather than policy reasons. We would like to provide an architecture that facilitates the use of intermediaries to allow communication between entities that for whatever reason cannot communicate directly. In Figure 2, hosts on the right-hand side requesting resources located in the private network on the left-hand side should be able to access the resources, provided forwarders $F1$ and $F2$ can communicate and maintain a persistent connection to each other.

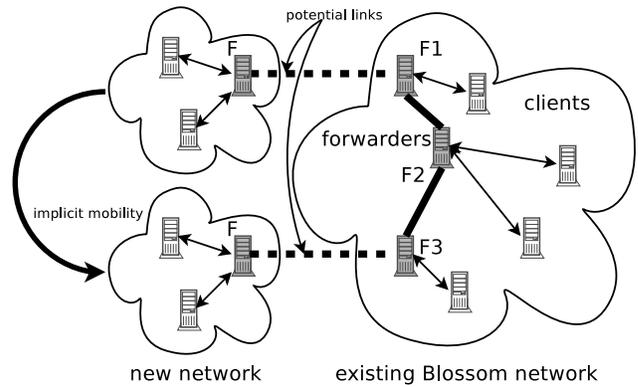


Fig. 3. DISTRIBUTED MANAGEMENT. Adding a network and its abundance of resources to the system need not require specific allocation of names, addresses, or routing from centralized authorities.

Distributed Management. Adding a network and its abundance of resources to the system need not require specific allocation of names, addresses, or routing from centralized authorities. Contrary to popular belief, the Internet is not entirely a distributed network. While its management is somewhat decentralized, many aspects of its structure and governance are hierarchical in nature. Autonomous systems engage in peering relationships in a manner that promotes the set of “tiers” that characterize the organization of Internet service providers today. Both the addresses and the names used to identify resources are allocated by a collection of governance organizations, arranged hierarchically. Such an arrangement is contrary to the underlying relationships among organizations interested in using the Internet to communicate. In some sense we would like to provide a means by which the Internet can grow without requiring the consent of far-removed third parties. In Figure 3, a new network fragment on the left is set up to deploy a Blossom forwarder called F . Adding this fragment to the existing Blossom infrastructure requires only that a persistent connection be established with an existing Blossom forwarder. In this case, forwarder $F1$ might be chosen initially, but if later $F3$ becomes reachable or more convenient, the new forwarder F can set up a persistent connection with $F3$.

Deployability. Our system must be deployable within the current Internet infrastructure. Any complex system of sufficiently large scale that cannot be deployed incrementally will never amass enough interest to overcome the economic hurdles to deployment. Our system must provide substantial benefit even if its rate of adoption is quite limited. So, we require that our system can coexist with existing Internet infrastructure. In particular, both clients and servers should be able to easily use both our system and the underlying Internet architecture.

Throughout our analysis, we consider resources to be identified by names meaningful to humans (in this case, hostnames provided by DNS), rather than by their underlying network-layer addresses. This provides the advantage that the mapping of IP addresses to names can occur via local DNS.

Note that the Blossom design achieves its seemingly conflicting goals of locality and universal access at the expense of universal naming. Indeed, one of the key features of the Internet today is that names used to identify resources are universal: they depend only upon the resource and are not defined by the name, physical location, or logical location of the entity requesting the resource. We argue that universal naming is not indispensable, and we believe that by relaxing this constraint we can achieve a considerably more flexible network.

IV. ARCHITECTURE

Blossom consists of a peer-to-peer overlay network of *forwarders*, each of which has access to some set of Internet resources. Some resources may be available to some nodes but not others. The overlay network that connects all of the forwarders to each other includes a *data plane* that carries tunnelled DNS requests and TCP sessions, as well as a *control plane* that carries routing information.

We believe that it is possible and beneficial to design a system that sacrifices globally apportioned names in favor of a distributed method of assigning names. Hence, Blossom allows for the existence of multiple independent Internet fragments, possibly overlapping, each with its own set of names. By considering each fragment to be its own namespace, we avoid requiring that all names are globally unique. However, in order to achieve universal access, we must provide a means by which all resources can be named. To this end, we stipulate that that names of forwarders are globally unique, and we identify some target resource R as a combination of the name of a forwarder that can reach R concatenated with the name of R as seen by that forwarder. Unlike Internet hosts, Blossom forwarders *choose their own names* by generating a self-certifying identifier and using that as a global name. In this sense, each resource accessible via Blossom is associated with at least one globally unique name, but the resources themselves are not responsible for guaranteeing global uniqueness—instead, all that is required is uniqueness within the local fragment observed by some particular forwarder.

A. Components

The Blossom system consists of a set of components that manage an underlying proxy network:

- **BLOSSOM FORWARDER.** Forwarders are the nodes in the peer-to-peer overlay network, working to establish circuits through which TCP streams flow.
- **BLOSSOM CLIENT.** The Blossom client consists of two components: (a) a proxy that serves as an intermediary between client applications and the overlay network, and (b) a mechanism for choosing paths and establishing circuits through the overlay network.
- **BLOSSOM DIRECTORY SERVERS.** The directory servers obtain information about the individual forwarders. Clients contact the directory servers in turn to obtain information necessary to route traffic to the forwarders of interest.

- **APPLICATION-SPECIFIC BLOSSOM PROXY.** Each client runs application-specific proxies as needed, which attempt to reformat hostnames and IP addresses in the application-layer datagram into the format that clients and servers would expect to get during a regular session.

From a high-level perspective, Blossom forms an overlay network for TCP. Applications treat the Blossom client as a generic transport-layer proxy; this proxy may use the SOCKS [16] protocol. The Blossom client receives information about the status of the Blossom network via the directory servers and passes traffic to the network of Blossom forwarders, which ultimately complete the connection to the target server.

B. Example Applications

Next, we provide a few specific examples of scenarios in which Blossom may provide a (partial) solution to the problems caused by Internet fragmentation.

- **SIMPLE NAT/FIREWALL CIRCUMVENTION.** In this scenario, a user wanting to provide services administers an Internet peer positioned behind a NAT or a firewall intended to be protective. Blossom should be able to provide the user with her desired functionality without removing the NAT or firewall.
- **EVASIVE SIMPLE CONTENT FILTERS.** Whether it is the network providing the filtering or the service itself, it is often useful for a client to be able to see a service from the perspective of a peer somewhere else in the Internet. For example, it may only be possible for a client to access some resource if it appears to be sending requests from a particular address range or if geolocation indicates that it appears to exist in some specific country.
- **EVASIVE LARGE-SCALE BLOCKING.** Given a set of proxies that individually are ephemeral but are persistent when considered in aggregate, Blossom should be able to provide access to resources within highly-censored national networks, such as those of China, Saudi Arabia, or Iran. Similarly, Blossom should be able to provide access to Internet resources for users within such networks.
- **VIRTUAL PRIVATE NETWORKS.** Since Blossom provides an encrypted connection between the client and the Blossom last-hop forwarder, it may be possible to use Blossom as a simple VPN, for example in order to provide access to resources within a corporate network [10]. One could imagine providing an authentication service to run on the same machine as the Blossom last-hop forwarder.

V. IMPLEMENTATION

Our test implementation uses *Tor* [7], an onion-routing overlay, as a substrate. Blossom leverages the source-routing framework and general directory paradigm provided by Tor to create what might be considered an “extended” Tor network that satisfies the Blossom design objectives described in Section III.

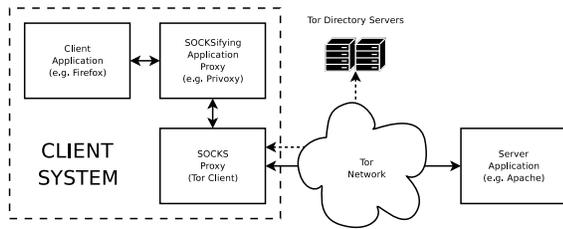


Fig. 4. CLIENT PERSPECTIVE DIAGRAM: TOR. Overview of Tor as it is presently deployed.

A. System Overview

Figure 4 depicts the Tor architecture from a client-side perspective. Tor consists of three main components: (a) a *client*, (b) a network of *forwarders*, and (c) a small number of *directory servers*. The Tor client periodically contacts the directory servers to ascertain a list of forwarders in the Tor network. On the application side, the client uses a SOCKS proxy to channel application data to the destination server via its network of forwarders. Often clients need an application-specific proxy (e.g. *sockat* or *Privoxy*) to make use of the SOCKS proxy. The Tor architecture has the characteristic that all forwarders must be mutually reachable; this is by design and intended to promote specific anonymity goals.

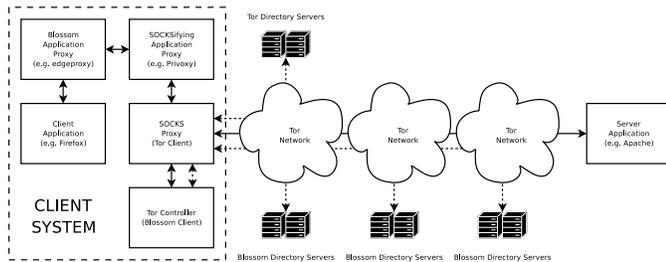


Fig. 5. CLIENT PERSPECTIVE DIAGRAM: BLOSSOM. Blossom consists of a set of components that manage an underlying proxy network. Our test implementation uses *Tor* as a substrate.

Figure 5 shows how Blossom changes this picture. Blossom provides (a) its own *client*, which manages the Tor client via a control protocol, (b) its own *directory servers*, which allow the publication of information about Blossom forwarders, and (c) application-specific Blossom proxies, which allow applications to take advantage of Blossom notation. The Blossom forwarders themselves are at their core ordinary Tor forwarders, except that since Blossom clients and directory servers manage how Blossom forwarders are used, there is no stipulation that forwarders must be mutually reachable. Hence, multiple Tor networks may be “bridged” via Blossom.

Blossom directory servers publish four different kinds of entries:

- **FORWARDER DESCRIPTOR.** Like Tor directory servers, Blossom directory servers provide *descriptors* that can be used by the Tor client to establish circuits through the forwarding network. Descriptors are self-signed statements

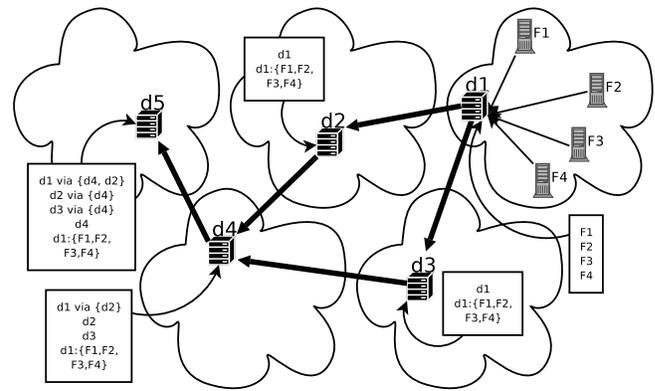


Fig. 6. ADVERTISING BLOSSOM ROUTERS. Blossom directory servers use a *path-vector* algorithm to propagate contact information for Blossom forwarders. Black lines indicate the path taken by an advertisement initiated by the directory server labeled *d1*.

published by forwarders that contain contact information, including IP address, port, and RSA key, as well as salient information about the capabilities of the forwarder, including exit policy and bandwidth measurements.

- **FORWARDER PATH.** Suppose that a Blossom forwarder publishes its descriptor to some particular directory. The Blossom architecture allows forwarders to publish their descriptors in directories in locations from which those forwarders are not directly accessible. If the forwarder is not directly accessible by nodes that receive descriptors from this directory, then the forwarder must provide instructions by which some client can reach it. These instructions appear in the form of a *path*, listing a particular sequence of nodes to which to connect to establish a circuit including the target forwarder. If, in the context of Figure 6, *F1* had published to *d5* directly, then there would be a forwarder path entry for *F1* describing how to get to *F1* from the vicinity of *d5*.
- **DIRECTORY TABLE.** Directory servers publish a list of all other directory servers in the system, as accrued over time through routing advertisements. Entries for directory servers that are directly reachable are trivial, containing only the name of the server. Other entries include a path through the set of directory servers via which the remote directory service may be reached. The first four entries in the box corresponding to *d5* in Figure 6 represent directory table entries.
- **DESCRIPTOR MAP.** Not all Blossom directories publish descriptors for all Blossom forwarders; however, given the name of a particular Blossom forwarder, every Blossom directory must know how to find the descriptor for that forwarder. Each directory server publishes an entry corresponding to each foreign directory server, with a list of Blossom forwarders whose descriptors are published at that directory server. The last entry in the box corresponding to *d5* in Figure 6 represents a descriptor map entry.

The directory servers propagate reachability information about individual entries (both forwarders and directory servers) in their respective databases to other directory servers throughout the system. In this manner, any client using any of the directory servers throughout the system will have a measure of assurance that its data will be routed to the requested forwarder. Figure 6 illustrates the process in which route information is propagated through the system. Entries are propagated using a BGP-like path-vector protocol, which includes a simple route selection protocol run at each of the directory servers.

B. Accessing Resources

Suppose that the forwarders have organized themselves into an overlay that can route TCP traffic. Each forwarder independently generates a self-certifying identifier, and forwarders throughout the system refer to other forwarders using these identifiers. As long as the size of the identifier is sufficiently large and the sources of randomness are sufficiently effective, the chance of a namespace collision among these identifiers within the system will be negligible.

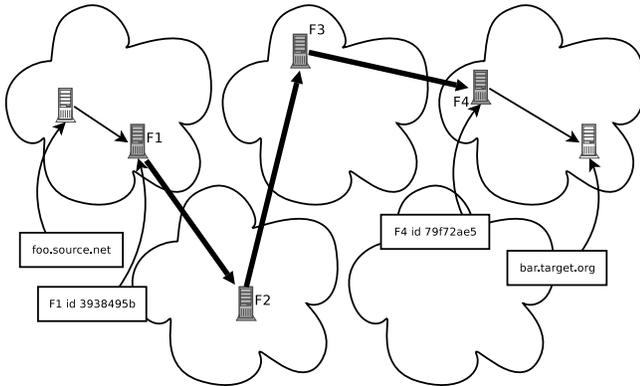


Fig. 7. ACCESSING A RESOURCE. The source establishes a connection to `bar.target.org.79f72ae5.exit`. DNS requests and TCP sessions are both tunneled through the infrastructure.

Next, we show how Blossom enables an Internet host to access resources outside its local fragment. See Figure 7. Suppose that the source (labeled `foo.source.net`) wants to communicate with a host known to forwarder F_4 as `bar.target.org`. Suppose that the source knows how to talk to F_1 , and that the self-chosen ID of F_4 is `79f72ae5`.¹ Then, the source will tell F_1 to open a TCP session to `bar.target.org.79f72ae5.exit` on its behalf. The control plane provides F_1 with routing information indicating that F_2 is the next hop en route to F_4 , so F_1 knows how to forward packets through the overlay to F_4 . Next, F_1 forwards the request for `bar.target.org` through the overlay to F_4 , who uses DNS to resolve it to an IP address. At this point, F_1 can tunnel the entire TCP session through the overlay to

¹We chose four bytes to create an illustrative example; actual IDs would be longer. Also, in practice we use human-readable names, mapped to self-certifying IDs by a third party.

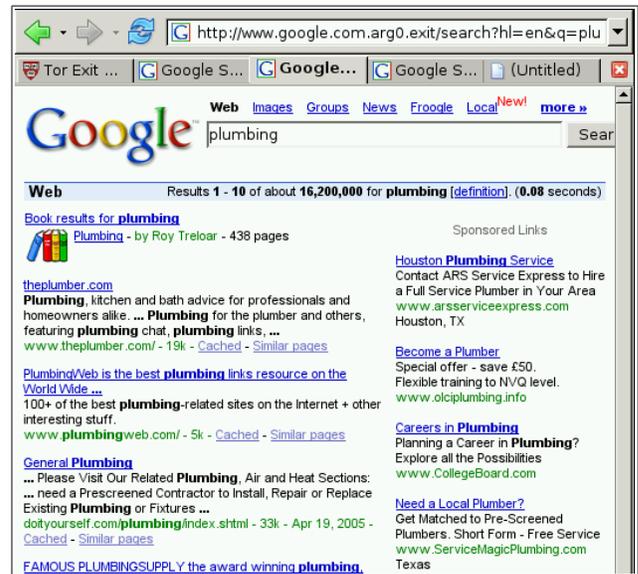


Fig. 8. A RESOURCE AS SEEN FROM DIFFERENT FORWARDERS. Consider the subtle differences in the Google search result page when viewed from two different forwarders. In this example, one forwarder (`justanothernode`) is located in New Hampshire and the other (`arg0`) is located in Texas.

F_4 . Note that this involves segmenting the TCP session—the conversation between the source and F_1 will have a different pair of source and destination addresses than the conversation between F_4 and the target resource. This means that Blossom will not work with end-to-end address-based security systems such as IPsec; we describe the policy implications in more detail in Section VI.

Observe that the combined name `bar.target.org-79f72ae5.exit` is globally unique, but the name was not apportioned by any authority of global scope. Also, there is no requirement that each resource is associated with exactly one forwarder; multiple forwarders may be able to reach the same resource, possibly using different names.

C. Evaluation

Next, we demonstrate the capabilities of Blossom in terms of *usefulness* (showing that Blossom is useful in practice today) and *performance* (showing that Blossom does not introduce too much unnecessary overhead).

1) *Usefulness*: Blossom allows clients to access resources that are not directly accessible. A simple example might involve showing that some resource is available when viewed from one forwarder and is not available in other cases. A more striking example, however, is to show that some resource is subtly different when viewed from one forwarder than it is when viewed from another forwarder. Figure 8 shows how the same web page may represent a different resource when viewed from different forwarders. In this example, the set of sponsored links advertised by Google differs depending upon the geographic locale of the host making the request. Generally, such differences result when either (a) the network forwards requests to a different server or (b) the server presents a different result based upon the address of the requester.

2) *Performance*: Next, we compare the performance achieved when accessing a resource via the Blossom network to the performance achieved when accessing the same resource via a set of carefully deployed SSH tunnels. As an experiment, we send files consisting of random bits from a particular source host S to a particular target host T . The graphs shown in Figure 9 compare the relative performance of transferring files of varying sizes via the following methods:

- VIA SSH TUNNELS. If the resource were not directly accessible, we would need a tunnel to reach it. To simulate this, we introduce a third host, F , whose purpose is to forward data from S to T . We establish two SSH tunnels, using the same cipher that Tor uses (`aes128-cbc`), one from F to T and one from S to F in which the endpoint of the SSH connection is T . Files are sent to the local port on S that forwards the traffic through both tunnels en route to T .
- VIA THE BLOSSOM INFRASTRUCTURE (ONE HOP). S runs a Blossom client, and T is a Blossom forwarder directly reachable from S . S sends files via Blossom to `localhost:port` via T , which means that files are sent to a SOCKS proxy on S , which forwards them through Blossom to T , which passes them to `localhost:port` (on T).
- VIA THE BLOSSOM INFRASTRUCTURE (TWO HOPS). This case is the same as the one-hop case, except that we establish F and T as Blossom forwarders such that traffic sent from S to T via Blossom must be carried by a circuit that traverses F .

We conclude based upon the performance ratio that while Blossom introduces some overhead that SSH tunnels do not,² Blossom is still feasible as a real-world solution to the problem of not being able to access certain resources directly. Our

²Notably, Blossom incurs overhead associated with (a) setting up a Tor circuit and (b) using the circuit, which employs a fixed window size designed for a general case that assumes heavy network usage with many streams.

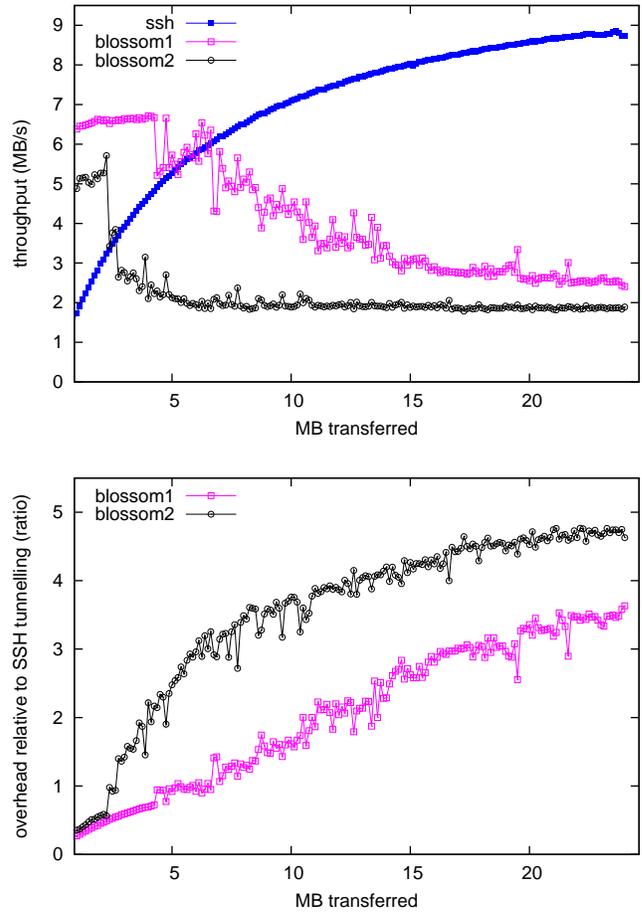


Fig. 9. FILE TRANSFER PERFORMANCE. We evaluate Blossom performance by assessing the time needed to transfer a file of a particular size through the Blossom forwarding framework. The top graph compares the relative throughput of transferring the file (a) via SSH tunnels, (b) via Blossom with a path length of one, and (c) via Blossom with a path length of two. The bottom graph presents the overhead introduced by Blossom compared to SSH tunnelling in terms of a ratio.

estimates are conservative; we are presently working with Tor developers to improve the performance of the underlying infrastructure. We further conclude that Blossom eliminates much of the difficulty associated with establishing specific tunnels corresponding to specific services *ex ante*—Blossom allows clients to “see the world” from the perspective of a forwarder, restricted only by the forwarder’s policy, without establishing special tunnels in advance.

VI. REMARKS

There are two fundamentally distinct views about how to improve trust in modern computer networks. The first view accepts that entrenched application-layer protocols are insufficiently functional to provide the necessary security properties. The logical solution, the reasoning concludes, is to implement authentication at lower layers in the protocol stack. The preponderance of interest in IPsec [14], particularly in the context of IPv6 [6], provides an excellent demonstration of

the perceived need for network-layer authentication. Combined with a means of binding addresses to identities, we can know the identity of our interlocutor simply by virtue of its network-layer address and authentication information. The grand vision is that network layer addresses are ordained from a central authority, in a hierarchical manner, such that there is a clear path leading to a specific individual or group responsible for each individual address in the system.

The second view asserts, in accordance with the end-to-end principle [19], that locators (addresses used by IP routing systems to direct traffic to a destination) are really only for routing datagrams. The Internet itself merely provides transport for applications, and the applications themselves are responsible for their own security properties. Dynamic addressing, NAT, and proxies ultimately limit the usefulness of the practice of binding routing addresses to identity. Protocols that provide authentication below the transport layer (e.g. IPsec, WEP, etc.) have important uses, but in general, such uses are not a substitute for end-to-end authentication. A world of globally unique, hierarchically ordained routing addresses intended for use as identifiers, according to the logic of this argument, is neither feasible nor desirable.

The Blossom design finds itself squarely in line with the second view. The ideal Blossom-enabled Internet would be opaque: if A and B are both connected to the Internet, then either can start a conversation with the other. Ideally, the conversation would be anonymous to the network as well. However, we recognize that there are serious policy issues surrounding anonymity, and we do not address those issues here.

VII. CONCLUSION

Our work offers two main contributions. First, we provide a set of design goals that afford a new way of considering Internet resources, specifically the idea of sacrificing globally apportioned names to allow locality in naming, distributed management, and universal access to resources. Second, we propose and have built Blossom, a peer-to-peer overlay of forwarders that makes it possible for hosts to communicate across multiple, independently managed Internetworks, despite each having its own independent naming and routing infrastructure. Blossom can be used for a number of applications including circumventing censorship-based content filtering and virtual private networks. Moreover, unlike previous related systems, Blossom requires no changes to client and server applications.

Some technical issues that we have not examined in this paper include (a) the performance and robustness of the algorithm for propagating forwarder information throughout the network of directory servers, (b) identifying forwarders by category or human-readable names, and (c) how to provide continuous connectivity between fragments in which the links are always changing. We believe that there are a number of interesting questions for future study within all of these areas.

Important policy questions exist as well, since Internet peers may gain access to some resources to which they did not have access previously. Routing around network-layer access

restrictions has various implications, and we will have the task of showing that our system can coexist with reasonable policy frameworks. Another question is whether Blossom can be used to resolve competition for the allocation of resource names.

There are both costs and benefits to considering a less rigid, more organic structure to the allocation of resource names. We believe that this work presents a new and useful way of considering the organization of Internet services and Internet connectivity.

REFERENCES

- [1] L. Abba, M. Buzzi, D. Pobric, and M. Ianigro. Introducing transparent web caching in a local area network. In *Proceedings of the 26th International Computer Measurement Group Conference*, 2000.
- [2] S. Bhattacharjee, K. L. Calvert, and E. W. Zegura. Active Networking and End-to-End Argument. In *ICNP*, 1997.
- [3] M. Bright. BT Puts Block on Child Porn Sites. *The Guardian*, 6 June 2004, 2004.
- [4] D. R. Cheriton and M. Gritter. TRIAD: A New Next-Generation Internet Architecture. <http://www-dsg.stanford.edu/triad/>, 2000.
- [5] D. Clark, R. Braden, A. Falk, and V. Pingali. FARA: Reorganizing the Addressing Architecture. *ACM SIGCOMM Computer Communication Review*, pages 313–321, 2003.
- [6] S. Deering and R. Hinden. Internet Protocol, Version 6 (IPv6) Specification. Internet Engineering Task Force: RFC 2460, 1998.
- [7] R. Dingleline, N. Mathewson, and P. Syverson. Tor: The Second-Generation Onion Router. In *USENIX Security Symposium*, 2004.
- [8] N. Feamster, M. Balazinska, G. Harfst, H. Balakrishnan, and D. Karger. Infranet: Circumventing Censorship and Surveillance. In *Proceedings of the 11th USENIX Security Symposium*, 2002.
- [9] B. Ford. Unmanaged Internet Protocol. In *HotNets*, 2003.
- [10] B. Gleeson, A. Lin, J. Heinanen, G. Armitage, and A. Malis. A Framework for IP Based Virtual Private Networks. Internet Engineering Task Force: RFC 2764, 2000.
- [11] G. Goodell, W. Aiello, T. Griffin, J. Ioannidis, P. McDaniel, and A. Rubin. Working Around BGP: An Incremental Approach to Improving Security and Accuracy of Interdomain Routing. In *Proceedings of the Network and Distributed System Security Symposium*, 2003.
- [12] T. G. Griffin, F. B. Shepherd, and G. Wilfong. Policy Disputes in Path Vector Protocols. In *Proceedings of the Seventh International Conference on Network Protocols (ICNP 1999)*, 1999.
- [13] T. G. Griffin and G. Wilfong. On the Correctness of IBGP Configuration. In *Proceedings of the ACM Conference of the Special Interest Group on Data Communication (SIGCOMM)*, 2002.
- [14] S. Kent and R. Atkinson. Security Architecture for the Internet Protocol. Internet Engineering Task Force: RFC 2401, 1998.
- [15] S. Kent, C. Lynn, and K. Seo. Secure Border Gateway Protocol (Secure-BGP). *IEEE Journal on Selected Areas in Communications*, 18(4):582–592, 2000.
- [16] M. Leech, M. Ganis, Y. Lee, R. Kuris, D. Koblas, and L. Jones. SOCKS Protocol Version 5. Internet Engineering Task Force: RFC 1928, 1996.
- [17] R. Mahajan, D. Wetherall, and T. Anderson. Understanding BGP Misconfiguration. In *Proceedings of ACM SIGCOMM 2002*, pages 3–16. ACM, 2002.
- [18] Y. Rekhter and T. Li. A Border Gateway Protocol 4 (BGP 4). Internet Engineering Task Force: RFC 1771, 1995.
- [19] J. H. Saltzer, D. P. Reed, and D. D. Clark. End-to-End Arguments in System Design. *ACM TOCS*, 2(4):277–288, 1984.
- [20] A. C. Snoeren and B. Raghavan. Decoupling Policy from Mechanism in Internet Routing. In *HotNets*, 2003.
- [21] J. Stewart. BGP4: Interdomain Routing in the Internet. Addison-Wesley, 1998.
- [22] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, and S. Surana. Internet Indirection Infrastructure. In *Proceedings of ACM SIGCOMM*, 2002.
- [23] K. Varadhan, R. Govindan, and D. Estrin. Persistent Route Oscillations in Inter-Domain Routing. *Computer Networks*, 32(1):1–16, 2000.
- [24] M. Walfish, H. Balakrishnan, and S. Shenker. Untangling the Web from DNS. In *NSDI*, 2004.
- [25] M. Walfish, J. Stribling, M. Krohn, H. Balakrishnan, R. Morris, and S. Shenker. Middleboxes no longer considered harmful. In *OSDI*, 2004.