Rethinking Home Networks in the Ultrabroadband Era

Michael Rabinovich*, Mark Allman†, Stephen Brennan*, Brian Pollack*, and Junbo Xu*
*EECS Department
Case Western Reserve University
Cleveland, OH, USA
{michael.rabinovich,stephen.brennan,brian.pollack,junbo.xu}@case.edu
†International Computer Science Institute
Berkeley, CA, USA
mallman@icir.edu

***Abstract*—The advent of ultrabroadband Internet connectivity brings a 2-3 orders of magnitude jump in the capacity of access networks (a.k.a. the "last mile"). Beyond mere capacity increase, this leap represents a qualitative shift in the overall Internet environment. Therefore, we argue that only by seizing the opportunity to re-think the way we structure network applications and services can we realize the full potential ultrabroadband provides.**

Specifically, with ultrabroadband residential networks, we have the opportunity to re-center our digital lives around our residence, similar to how our physical lives generally center around our homes. To this end, we introduce a new appliance in home networks–a "home point of presence"–that provides a variety of services to the users in the house regardless of where they are physically located and connected to the network. We illustrate the utility of this appliance by discussing a range of new services that both bring new functionality to the users and improve performance of existing applications.

***Index Terms*—network architecture, home networks, ultrabroadband Internet**

I. Introduction

People generally center their lives around their residence. This *center of gravity* is where we can be contacted, store our things, do our homework, play games, meet after disparate activities, eat our meals, and so on. At times we branch out from our homes for specific tasks. We may rent a storage shed for some items that we do not have room to store. Or, we keep the bulk of our money in a bank because of the added protection it enjoys at such institutions. Or, we may eat out for fun or because we do not have time to cook. However, while we branch out for specific reasons, our homes are always the hub of our lives. While we do not always view it as such, this central hub in our lives *facilitates choices*. For instance, it offers a place where we can get our tools out and change the oil in our car, as opposed to going to a mechanic to have this service performed. Or, we can use our own kitchen to prepare a meal precisely as we wish rather than going out and getting a dish that is a little too spicy or not spicy enough. We make such choices every day without giving them a second thought.

This work was supported in part by NSF through grants CNS-1647145 and CNS-1647126.

Our digital lives are not organized around such a hub. Rather, we use a myriad of services to communicate with one another, store pictures, work on documents, share videos, keep our music, deal with calendars, etc. In this arrangement *we* are the hub. Our content and information comes to us from a range of places to wherever we happen to be at the moment. This user-centric arrangement clearly has its benefits. No longer are we beholden to remembering to take a physical photo album to lunch with a friend to show them pictures of our kids. Rather, we can just display these on our phone or tablet at a moment's notice. The flip side of ubiquitous access to our information is a more distributed footprint in the network with a user's information strewn around a variety of services. This naturally leads to a case where users are not fully in control of their own information, but rely on trusting a bevy of service providers to keep their information private and safe. For instance, rather than keeping our medical records in a shoebox under the bed, this new model calls for us to keep them somewhere out in "the cloud" where we can always access them. However, at that point we are not the only ones who can access the records. Further, in an ironic twist, this "ubiquitous access" may in fact be a heavy burden if we become incapacitated and a relative needs to help a doctor understand our medical history and cannot access the information. Finally, our personal information is dispersed among numerous services—Google docs in one place, Facebook profile in another, photos elsewhere, email elsewhere again. Maintaining and accessing information across this dispersed footprint becomes increasingly difficult.

Conceptually, of course, we could organize our digital lives around the same hub as we organize our physical lives: our residence. We believe there are two major reasons why we have not taken this approach. First, providing ubiquitous access to information stored in our home is problematic given the capacity of today's home networks. Second, sharing information from within one's house requires running servers, controlling who can access what data, setting priorities for information access and retrieval and fixing issues as they arise. These skills are beyond the vast majority of the population.

However, we are witnessing a rapid increase in bandwidth

of residential networks. Bandwidth of tens of megabits per second has already become routine, and now fiber-to-the-home (FTTH) is on the verge of making revolutionary new capacity available. For example, Google has made gigabit residential connectivity available in a number of cities across the U.S. [1] and Chattanooga's power utility has connected homes in its service area via fiber [2] and has made 10Gbps connectivity available to each of the 175K households in the 600 sq. mile area [3].

Modern high capacity networks remove one of the large barriers for people to center their digital lives around a hub in their home. Rather than having a small pipe to share content, users of these networks will have a nearly endless supply of capacity. While increased capacity is useful in itself, we argue that ultrabroadband opens the door for re-thinking how we structure people's digital lives. Specifically, we discuss an approach that would provide users with applications that offer both ubiquitous access and global reach in a form that is as convenient as using external services. This approach promises to allow users to retain control of their data and network footprint while improving performance of Internet communication. Our central theme involves creating a *home point of presence* (HPoP) that can serve as the hub for a given household. Our goal in building such an appliance is to abstract many of the technical details of operating such a system from users, while still allowing them to retain control of their own resources. Additionally, we note that we are not trying to preclude cloud-based systems. Rather, we view an HPoP as an option that users may exercise in some cases and not in others. As sketched above, this is analogous to other aspects of life (e.g., kitchens do not obviate the need for restaurants). Furthermore, as discussed later, HPoP brings about a hybrid choice whereby cloud-based applications can access—but not retain—data stored in HPoPs.

II. The Vision

A gigabit last mile for residential users brings new realities into the global network environment, which we argue warrants new thinking in how network applications should be structured and the information and network services should be organized and developing new understanding about the operation of our standard protocols within these high-capacity last mile environments.

Being content with the benefits purely from the increased capacity of access network, without rethinking the current organization of network applications, would dramatically underutilize the opportunities provided by ultrabroadband. Prior work delivers a stark evidence in support of this claim [4]. This study measured communication performance on Case Connection Zone (CCZ), an experimental FTTH network run by Case Western Reserve University that connects a small neighborhood adjacent to campus of roughly 100 residences via bi-directional 1 Gbps fiber links. By considering data transfer rates in each second of communication, this study found that CCZ users only exceed a download rate of 10Mbps 0.1% of the time and a 0.5Mbps upload rate 1% of the time. Clearly these users do not fully benefit from the capacity of their access networks. We identify the following key new realities due to the gigabit last mile.

Capacity Increase: The 2–3 order of magnitude increase in edge network capacity is a dramatic jump and in the short term likely leapfrogs both our applications' ability to use the capacity and the increases in capacity of the rest of the infrastructure—i.e., the Internet servers and core network. Therefore, at least for the near term, a 1–10 Gbps residential connection can be viewed as having infinite last mile capacity, with other parts of the infrastructure determining the quality of user experience on the Internet.

Bottleneck Shifts: The capacity increase at the last mile link can lead to bottleneck shifts. While the last mile with FTTH may have seemingly unlimited capacity, the bottlenecks will manifest in other locations. As a simple example, in Case Connection Zone (CCZ) [5] each home is served by a 1 Gbps link, but the roughly 100 homes are then immediately aggregated onto a shared 10 Gbps link by the service provider. It is likely that there will be periods when the aggregate link will become the bottleneck, which is different from the currently common case of the last mile being the bottleneck. And, while the last mile capacity is increasing, no commensurate change in core networks and server resources is expected. Thus, the bottlenecks may well shift back towards the middle of the network.

Lateral Bandwidth: A property related to the issue of shifting bottlenecks is plentiful lateral bandwidth to one's neighbors. For instance, the CCZ users have dedicated 1 Gbps connectivity to each other, bypassing any upstream bottlenecks discussed in the previous paragraph. In essence, a gigabit neighborhood becomes analogous to todays' enterprise or data center networks in terms of enabling applications that are not generally used over the WAN. Considering multiple such FTTH neighborhoods of the future, this creates a hierarchy of connectivity. A host has access to its local devices connected with, e.g., Firewire S3200 or USB 3 at 3–4Gbps, to its peers within the FTTH community at 1Gbps, and to the rest of the Internet through the shared aggregation link. Networked applications can benefit from factoring this hierarchy into their design.

Always-on connectivity: The switch from dial-up to today's broadband access networks signified a shift from intermittent to always-on connectivity. However, by combining always-on connectivity with the other changes mentioned above, ultrabroadband enables different thinking about Internet applications and services, which we argue will bring qualitative new benefits to our digital lives.

Obviously these new realities constitute a significant change. Therefore, the key question we pose is: *How can we re-think the organization of networks, applications and data to capitalize on the new opportunities provided, and to reflect the new realities introduced, by ultrabroadband networks?*

Our general thesis is that the dramatic shift in relative capacity between home networks and the rest of the Internet warrants a corresponding shift of the center of gravity

for users' data and activities towards their own homes. We argue that this will allow both quantitative (performance) and qualitative (new functionality) improvements in end-user applications and thus in user experiences.

To realize this shift, we envision a piece of equipment within an ultrabroadband home network that maintains a fixed presence on the Internet for its user(s). We call it a *home point of presence*, or HPoP. The HPoP functionality could be built into the home's access router and not require a distinct device, or co-locate with another resident device such as a set-top box or a home file server. In any case, we assume it is operational as long as there is power and online as long as there is Internet connectivity, regardless of which if any end-user devices are connected. In the rest of the paper, we expand on the general HPoP notion as a framework for new network services as well as for improved existing applications, and discuss some examples of these services the HPoP enables. In fact, to the extent that always-on connectivity is already exhibited by today's broadband home networks, some of our ideas, while spurred by our FTTH thinking, could prove beneficial even in today's residential networks.

Finally, we note that while we sketch a series of ways to restructure networks in a world where ultrabroadband is common, we do not mean to preclude other ideas. In fact, part of our goal in writing this paper is to spur the community to think about how networks may work differently to realize the promise of these ultra-high speed networks.

III. HPoP

We envision a "home point-of-presence" (HPoP) in each ultrabroadband-connected residence. We envision the HPoP as an extensible and configurable platform that can also run myriad mundane services for the user and the household—e.g., a contacts server, a calendar server, or an email inbox (in addition to various new services we discuss below). The HPoP provides seamless access to these services across various devices.

For the HPoP to facilitate centering users' digital lives around their residence, users must be able to connect to HPoP whether they are inside or outside of their homes. Thus, a preliminary issue that we must address is HPoP reachability in the presence of (potentially multiple levels of) address translation (NAT). For home networks that are behind a local NAT device only, the widely supported UPnP protocol [6] allows simple programmatic configuration for port forwarding as part of the HPoP setup. For those home networks that are behind ISP-operated NAT (co-called carrier-grade NAT, or CGNs), we assume the STUN protocol (or similar) [7] is used for this purpose. STUN works by so called "hole punching" through NATs, and not all NAT devices have the behavior required for hole-punching to work. In those cases, HPoPs can still be used, with limited functionality, employing relaying-based traversal mechanisms such as TURN [8]. IPv6 adoption continues to grow [9], therefore reducing IP address scarcity issues. This may diminish the use of NATs and make HPoP reachability straightforward. However, even if NATs persist, traversal techniques such as STUN and TURN will enable communication with HPoPs. In the rest of the paper, we assume that HPoPs are reachable from external networks for both UDP and TCP communication.

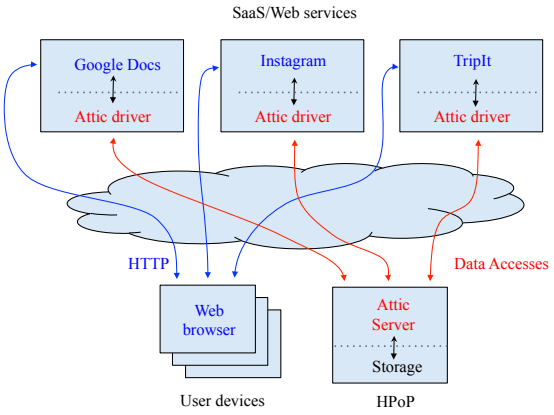

Fig. 1. A high-level view of a data attic architecture.

IV. HPoP Services

We discuss the benefits of an HPoP by describing a sample of useful services it can provide to future Internet users. Some of these services offer new functionality that would benefit users; others improve the performance of existing functions.

Note: This paper represents a synopsis of our vision of the potential of ultrabroadband networks and services. The following subsections include material from our previous work in striving to reach our vision. In particular, § IV-B includes material from [10], and § IV-C includes material from [11].

A. Data Attic

Modern Internet services often attempt to add value to users at the expense of their privacy. For instance, consider Google's suite of tools which enable users to work on and share everything from calendars to spreadsheets to email to pictures. These cloud-based applications bring great convenience to users by enabling users to operate on their data from anywhere, as well as readily share the data with others. Meanwhile, users do not incur the direct cost and maintenance issues that local software brings. However, the price for these conveniences is in giving up data and leaving it exposed to data mining by cloud operators and vulnerable to accidental leaks and other security breaches. In fact, even from the legal standpoint, there are weaker safeguards of one's data privacy if the data resides at a corporate data center vs. at a user's home [12]. Furthermore, this arrangment ties the user to the cloud provider in question as switching providers at best involves a cumbersome process of exporting the data from the old provider and importing it to the new one and at worst prevents one from wholesale data movement.

Our first sample HPoP service offers a new paradigm for organizing external applications, from Flickr and Facebook to cloud-resident software-as-a-service (SaaS) applications such

as Google Docs, and web-based email. Our approach calls for these applications to act on data stored a "data attic" in each user's home network instead of on a copy of the data that resides in the cloud. The data attic provides an application-agnostic interface to user data that external applications and services can access, but would not store or maintain the data outside that needed for a specific task. In a sense, the paradigm we envision is the inverse of Dropbox or Google Drive. As Figure 1 illustrates, a user's devices leverage the same HTTP interface to interact with the Internet services. However, these services obtain data from the user-run data attic instead of from their own data store. Obviously, denying SaaS providers control over user data will trigger changes in business models, with providers perhaps requiring subscription in lieu of the data possession. But data attics provide users with choice as to whether control of their data or a monetary cost is more important. We summarize key benefits offered by the data attics to users below.

- *User Control and Data Privacy:* A cloud provider aggregating data from millions of users represents an enticing target for hackers. As mentioned earlier, user data also has weaker legal protections when stored externally at a corporate data center [12]. Data attics leave control over data access entirely with the user. While individual home networks are less well managed and protected, data attics offer "defense in numbers", since they disperse user data among millions of home networks.
  The issue of user control over outsourced data has been on the radar of the Internet community. One of the only existing approaches that would withstand a security breach has the data self-destructing after use [13]. However, this notion complicates handling long-term data. Data attics sidestep the problem by retaining long-term control of the data. Another alternative would be to simply let the cloud store user data in encrypted form. The home network would then provide the external application the key to decrypt the data when an authorized user requests a particular service. The user would trust the application to not keep the key beyond the immediate use. While this indeed can help address the issue of data control, the data attic concept addresses additional issues—e.g., allowing changes and shared access by multiple actors, through multiple applications, while maintaining a single source for a file.
- *Provider Independence:* Data attics allow users to move freely between application providers that support the appropriate data formats.
- *Flexible Access:* As we have noted, users can access their data attic via cloud-based applications. Additionally, the data attic can act as a remote-disk and hence users can use their own local applications—such as word processors or spreadsheets—the work with their files. Further, just as some popular cloud-based applications have an "offline mode" (e.g., Google Docs [14]), similar use of attic-based data is possible. Just as with cloud-based applications, changes to the files would need reconciled upon re-connection (a plethora of approaches exist to address this issue, e.g., [15]–[17])

1) Case Study: Health Records: An especially intriguing application of the data attic is for aggregating the electronic health records of an individual (or family). These records are currently dispersed among providers, each requiring a separate release form, making a collection of one's full medical history hard (or impossible, e.g., when a past provider is no longer in business), especially in an emergency situation. Incomplete records may lead to delayed, duplicate, or even inappropriate procedures. Our approach will allow patients to easily aggregate and maintain their own electronic medical records in their home-based data attic. After a one-time bootstrapping with a provider to set up proper permissions—which would itself be largely automated—any records subsequently generated by this provider would be copied to the patients attic. The data then becomes portable across providers, and the patient can provide immediate access to their complete records as they see fit—e.g., circumventing complicated inter-institution protocols.

In this use case, the health record system at each provider would interact with each person's data attic. Further, each provider would retain a copy of the data to satisfy regulatory requirements. Therefore, the storage driver at the provider's site would duplicate writes to both local copy and the patient's remote attic.

The crucial difference between the data attic and current approaches to medical records aggregation such as Health Vault [18] or MedeFile [19] is that the latter tie the patient to the aggregation provider and require the patient to entrust the provider with data security. Just as with the cloud-resident data, these platforms represent an enticing target for the attackers because they concentrate data on millions of people. Meanwhile, a person's home only keeps a relatively modest amount of data about a handful of people.

Architecture: The architecture of the data attic needs to address the following key questions. First, how will external services be able to locate and access the required user data? Second, what is the proper layer at which the data attic should be implemented? In principle, it could be built based on the iSCSI interface as a remote disk, an HTTP server, or as a distributed file system such as NFS. Alternatively, the data attic may operate in different ways for different applications. Third, how will the attic be transparent to applications?

We have implemented a data attic prototype to investigate these questions. We chose HTTP(S) as the basis for our prototype and implement a data attic as a WebDAV server. HTTP(S) offers decoupled communication between the external applications and the attic and ease of firewall traversal. WebDAV further mediates access from multiple clients through file locking. While we envision the data attic as ultimately being an appliance that requires no nitty-gritty setup by users, our current prototype is a software system that runs on a general purpose computer. We have developed a script to automatically setup all portions of the system.

As an exemplar, we have developed an application that allows for the use of the data attic to aggregate health records. Before using a medical provider, the user gives the provider access to their data attic. This involves interacting with the local data attic portal to indicate the name of the new provider. At this point, the data attic will issue a QR code that includes all information needed to access the correct portion of the user's data attic—i.e., everything from the IP address of the data attic to the proper initial credentials to the location of the files within the attic[1]. The QR code is then furnished to the medical provider which in turn will link the user's data attic with the provider's medical records such that all records get pushed to the attic.

Finally, to interface with existing applications within the medical practice, our prototype replaces application's default *open*, *close*, *fopen*, and *fclose* function calls with our own function calls, implemented in the driver. We then recompile the applications using the "-wrap" option to the linker, which allows one to override an existing function. For example, any reference to "*open*" is replaced with a reference to "*__wrap_open*" that has the same input and output parameters but implements our own functionality—namely, a GET request for the file to the data attic. Upon receiving the file, the driver creates a local copy and opens it for the application. Subsequent accesses to the file will execute on the local copy, which will be sent back to the attic on close. No change to the application code is required.

Data Availability: Another potential challenge for data attics is data availability and preservation. Home networks are generally less reliable than large cloud data centers. For long-term data preservation, we can optionally backup the encrypted data locally—e.g., on a connected external disk or in-home NAS—or with a cloud such as Amazon Glacier. For data availability, users could either decide that occasional unavailability is an inherent reality of home utilities—similar to electric power—or add replication mechanisms. For instance, this latter may involve replicating the entire HPoP to attics belonging to friends and relatives, or redundantly encoding the contents—e.g., using erasure codes [20]—and storing pieces with a variety of peers.

B. Content Delivery with NoCDN

Scalable content delivery requires a widely distributed server platform with sufficient capacity to absorb temporary demand surges. Todays content providers either deploy such a platform themselves (e.g., Google) or use a third-party content delivery network (CDN) such as Akamai. Ultrabroadband affords the opportunity for an alternative approach to achieving scalable content delivery whereby content providers recruit well-connected users to allow their HPoPs to be effectively used as "edge servers" in an ad hoc CDN. The content provider could compensate in some fashion—e.g., monetarily or with free subscriptions to the content they serve. Users could serve content for multiple content providers. As content providers recruit users, a new content delivery marketplace directly between content providers and users—without a middleman in the form of a traditional CDN operator—will emerge.

While seemingly similar to a number of proposals for peer-to-peer CDNs (e.g., [21], [22]), the key difference with our approach is that previous systems use peers to *augment* a CDN platform, whereas we *eliminate* the third-party CDN altogether. We highlight this distinction by calling our approach *NoCDN*. While traditional CDNs offer a stable and low-risk way to improve performance and allow content providers to shed load, they bring non-trivial monetary costs to the content providers. The cost of traditional CDNs is commensurate with the resources these platforms bring to bear for their customers. The NoCDN approach leverages the spare resources users naturally possess in ultrabroadband edge networks. In particular, these spare resources are not part of business operations whereby costs and profits must be covered. Therefore, the cost to content providers will likely be lower than traditional CDNs. Further, there is flexibility in how the users are compensated in NoCDN. E.g., the New York Times may give free subscriptions to participants instead of direct monetary compensation. While this re-structuring offers promise in lowering costs and possibly giving better performance by serving content from closer to users than traditional CDNs, the structure also brings many research questions and risks, as follows.

Browser Transparency: Individual content providers generally lack the clout to compel users to change or reconfigure their browsers or install some sort of a download manager. Thus, NoCDN must work with unmodified browsers, which realistically means it must be fully implemented in standard JavaScript.

Content Integrity: Peer assistance in current peer-to-peer CDNs often comes with no awareness from the person who owns the resource. E.g., the agreement to act as a peer in Akamai's NetSession is buried inside the agreement to terms of use of various partners, such as Flash player or Autodesk [23]. Since users must explicitly sign up to become a peer in NoCDN, there is more danger that an attacker would sign up with an intent of corrupting the content that flows through the attacker-controlled equipment. Therefore, NoCDN must include mechanisms that ensure content integrity despite untrusted peers.

Accurate Accounting: One method of compensating peers is based on their objective contribution to content delivery—e.g., number of bytes served. Therefore, an unscrupulous peer has an incentive to inflate the contribution they report to the content provider. NoCDN must be able to protect content providers from such behavior.

Peer Selection: Traditional CDNs use some combination of edge server load and proximity to the client when choosing which content replica to use for a transaction. The methods are complex and constitute the CDN's "secret sauce". Without a traditional CDN to perform this operation, how should a content provider select a peer for the client to use? The

[1]Our current prototype does not handle the QR codes and requires the information to be manually entered at the provider's site.

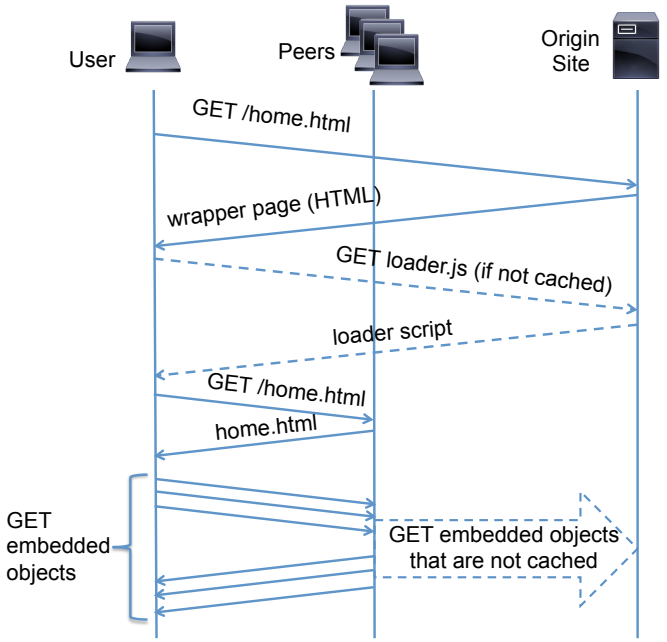

Fig. 2. Webpage access in NoCDN. The transfer of usage records to peers is not shown.

standard metrics that traditional CDNs lean on to select edge servers also apply in the NoCDN context—e.g., reachability, bandwidth, packet loss and delay. However, there is also a trustworthiness element to the decision—i.e., is the edge server faithfully carrying out its appointed tasks? Further, whereas traditional CDNs have general access to the edge servers and therefore can directly assess operation as needed, access to NoCDN nodes is likely to be more constrained. Therefore, investigating peer selection in this context is an open problem which will involve both the protocols for trustworthy telemetry to gather the relevant information about the peers and approaches for incorporating this information into peer selection decisions.

Leveraging Redundancy: Finally, we note that as the number of recruits for NoCDN content delivery increases, the scale offers optimization opportunities. For instance, the content provider could dictate that each object within a webpage come from a different source. Or, clients could download objects in chunks (e.g., using HTTP range requests) from disparate peers instead of as entire objects—as originally suggested in [24]. These options both spread the load and lower the chance that one problematic peer—be it malicious or overloaded—will have a large overall impact on the client.

To demonstrate the feasibility of NoCDN—and specifically the possibility of ensuring content integrity and accountability using a standard browser with no extra plugins—we have implemented a proof of concept prototype [10].

Our first observation is that traditional CDNs leverage loose handoffs within the system—e.g., via classic DNS request routing [25]—and largely allow various components to operate independently. This approach works because the components are under the CDN's control and therefore they can be trusted to perform their appointed tasks and accurately report the results (e.g., for billing purposes). However, NoCDN does not have the luxury of trust throughout the system and therefore loose handoffs leave room for abuse. Therefore, our prototype revolves around javascript that is served directly by the content provider and orchestrates the content retrieval and reporting functions.

The workflow of a page download is depicted in Figure 2. First, when a user accesses a content provider utilizing NoCDN, the content provider returns a *wrapper page*, which (a) lists the IP address of a peer from which to fetch the container object, (b) maps the URL for each recursively embedded object to the IP address of a peer to use for fetching the object, (c) includes a cryptographic hash of all page objects, as well as a unique short-term secret key for each peer listed in (a) and (b) above, and (d) includes a JavaScript *loader script* that fetches all objects from the peers, verifies the objects' hashes, assembles the objects into an integrated webpage and invokes the rendering function on the browser to display the page for the user.[2] Upon finishing the page download, the script transfers a usage record to each peer. The usage report is secured via a cryptographic signature using the secret key furnished by the content provider and includes a nonce to prevent replay. The NoCDN peers accumulate usage records and periodically upload them to the content provider for payment.

NoCDN does increase the chances of collusion attacks compared with traditional CDNs. In this case, a NoCDN peer and a client collude to download content—or claim to download content—for the sole purpose of coaxing payment from the content provider. The content provider can mitigate this by including some randomness in the client-to-peer mappings and hence making the payment path unpredictable, by using anomalous behavior detection or by non-usage-based payment methods (e.g., flat or capped payments, or non-monetary benefits as mentioned earlier).

Each NoCDN peer acts as a normal reverse proxy when processing user requests—i.e., the peer serves the requested object from its cache if available or, if not, obtains the object from the origin server, forwards it to the user, and caches it locally for future requests. Our prototype uses standard Apache in reserve proxy mode with virtual hosting—to allow a peer to sign up for content delivery with multiple content providers—as the peer implementation.

This mechanism improves scalability of the origin site because it only has to deliver a small wrapper page, with the loader script eminently cacheable and the rest of the page content fetched from the peer(s). The wrapper page would typically be generated dynamically since its contents reflect the peer selection for the user and includes short-term secret keys. Still, depending on the peer selection policies and billing models employed by the origin site, even the wrapper page

[2]Note, that the loader script is generic and can be cached by the browsers.

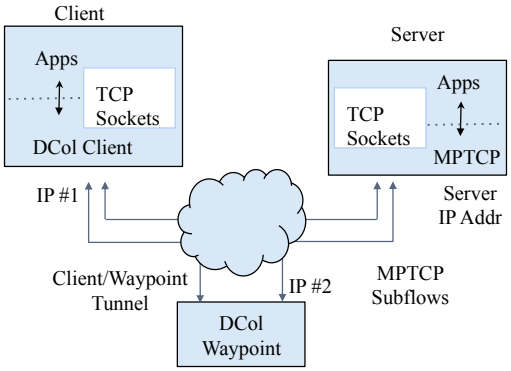

Fig. 3. Using MPTCP to create an application-transparent detour.

may be reused among users and/or allowed to be cached by the user for a certain time.

C. Practical Detour Communication

Numerous studies conclude that communication between two Internet hosts can be improved if the hosts forego the native path offered by IP routing in favor of sending traffic via a third-party relay [26]. The overlay *detour* paths produced by the relay hosts often have less packet loss [27], lower latency [28], and higher bandwidth [29], due to inefficiencies in native IP routes. Moreover, past studies have shown that most performance benefits can be obtained by using a single waypoint between the end-hosts [27], [30].

While the benefits of detour routing have been well established for almost two decades, the use of detours has largely been limited to communication within distributed platforms, notably, Akamai's "SureRoute" technology [31]. Numerous previously proposed overlay or peer-to-peer networks typically operate at the application layer and are not transparent to the application at both sides of the connection. Even when their functionality is encapsulated within a runtime library, such as in RON [32], applications still require modification to link to the custom library. However, ultrabroadband residential networks hold promise as locations for well-connected waypoints and hence potentially significant performance benefits to Internet users—especially if users can utilize multiple detour paths in parallel to speed up their content accesses.

To be practical, detours must meet several requirements. First, they should be transparent to applications at both ends of the communication, and especially to Internet servers that are outside clients' control. Second, because it is difficult to predict if a particular detour will be beneficial or harmful to a given communication, hosts should be able to add, remove, or change detours dynamically in the course of the communication. This would make it possible to select detours by using "trial and error" to explore multiple detours and retain the beneficial ones. Third, while some applications may provide point solutions for parallelizing communication through intermediaries at the application level, detours should be available to all applications.

We explore an HPoP-enabled approach satisfying the above requirements in [11]. Our approach—termed the "Detour Collective" (DCol)—calls for users forming cooperatives in which members agree to serve as waypoints to each other.

We leverage multipath TCP (MPTCP) [33] to make detours transparent to applications. Figure 3 illustrates our solution. MPTCP normally allows a device to communicate with a server on several network interfaces in parallel, but we use it to instead communicate through external waypoint hosts. In the figure, both the client and waypoint are part of the cooperative and install the DCol code. To engage a waypoint in its communication with the server, the client establishes a tunnel between the client and waypoint. The waypoint then mimics an MPTCP subflow to the server, making the server oblivious to the overlay detour. The server will not understand that the two subflows are not coming from two interfaces on the same device—as in regular MPTCP—but from two separate machines, potentially located in different corners of the world.

In the common case when most of the data flows from the server to the client, the client establishes subflows to the server through waypoints, but it is still up to the server to split the communication among the waypoint(s) and the original direct path to the client. However, the client can indirectly influence these decisions by withdrawing undesirable detours (i.e., closing the corresponding subflows), adding new detours, or manipulating the acknowledgements. Since the default MPTCP schedulers use RTT as a key factor in their subflow scheduling policy, a custom client's scheduler can reduce server's use of a detour by delaying subflow-level acknowledgments of the corresponding subflow and thus increasing the RTT values seen by the server. When the data flows mostly from the client to the server—e.g, in file uploads, video-conferencing, or other increasingly common cases of user-generated content—the client can directly explore different waypoints by sending a few data packets over new subflows and staying with whose waypoints that perform well. In either case, the client is able to explore different waypoints to find efficient overlay detours, and further to aggregate bandwidth of several available paths. Most importantly, since MPTCP presents the same binary-compatible OS-level API as TCP, unmodified applications may use this mechanism simply by using our patched kernel.

While very few content servers currently support MPTCP [34], there are signs that this might be changing. For instance, Apple uses MPTCP for its Siri service [35]. We assume that as compelling use cases such as the one offered here emerge, the deployment by content servers will grow. Additionally, the IETF is working on a proposal to facilitate deploying MPTCP proxies within the network. This approach allows MPTCP-adopting clients to benefit from MPTCP even when interacting with a non-MPTCP servers, by leveraging an MPTCP proxy in server's vicinity [36]. Our approach can be used in this deployment scenario as well, by establishing subflows with the MPTCP proxy.

Client-to-Waypoint Tunneling: Since both client and waypoint run DCol code, they can implement any custom tunnel-

ing mechanism. However, standard tunneling mechanisms are readily available for easy implementation. We have explored two existing mechanisms for tunneling MPTCP subflows between the client and waypoint in our prototype: VPN tunneling and network address translation at the waypoint. Our prototype can chose either technology for individual detours interchangeably.

With VPN tunneling, a client uses packet encapsulation to divert traffic through a waypoint. In our prototype, a waypoint runs an OpenVPN server with DHCP service for its VPN. To use a waypoint for detouring, the client creates a virtual interface and joins the corresponding VPN by running DHCP to acquire an IP address on the virtual subnet. The client further sets up a routing rule for this interface with high cost to all destinations in an effort to prevent the interface from being used for unrelated traffic. To connect to multiple waypoints simultaneously, the client would create multiple virtual interfaces and join multiple VPNs. To avoid addressing conflicts between the VPNs, each waypoint can use a distinct address block for its private IP subnet. For instance, consider assigning each waypoint in the collective a /26 from the 10.0.0.0/8 block of private addresses. This allows for each of 256K non-conflicting waypoints to serve 64 clients simultaneously. In our prototype we assign subnets manually, but in a large collective, subnet allocations would be managed by an appropriate management plane.

With NAT tunneling, the client and waypoint negotiate a port on which the waypoint would receive packets from the client and the intended final destination of those packets. This forwarding is realized via standard NAT translation rules. Then, the client's kernel—with DCol modifications—addresses packets to agreed upon port on the waypoint. The waypoint forwards these to the final destination, rewriting the IP addresses and port numbers accordingly. Replies are similarly passed from the waypoint to the client. Linux offers standard NAT facilities within the netfilter framework to set up the NAT rules, with no custom code needed for packet forwarding.

The two tunnelling techniques have their tradeoffs. Once a client establishes a VPN tunnel with a waypoint, this tunnel may be reused to create a detour for any TCP connection to any server, without any additional setup. The NAT mechanism requires signaling with the waypoint for every new server address and port number combination. On the other hand, VPN adds 36 bytes of per-packet overhead for IP encapsulation and UDP and OpenVPN headers, while NAT adds no extra bytes to a packet.

Security: Inserting a waypoint into a communication path introduces potential security and privacy concerns. However, modern Internet applications increasingly use TLS [37]. Our prototype requires the client to complete the TLS handshake with the server over the direct path before establishing any detours. Therefore, any subflows through detours will be encrypted. While this keeps the contents obscured from the waypoints, the waypoints still learn the IP addresses with which the client is communicating. While this information already flows openly through the Internet in the unencrypted IP headers, our approach makes it readily available to the waypoints involved. This is an inherent cost of DCol.

A malicious waypoint could also disrupt its subflow in arbitrary ways. For instance, by dropping some or all of the packets on the subflow. The application can detect the resulting performance impact and withdraw this waypoint, while transparently recovering the affected packets over the remaining subflows. Furthermore, the misbehaving peer can be expelled from the collective to avoid future issues.

D. Internet@home

As network and computer capacities grow, latency will inevitably become the performance bottleneck that limits the responsiveness of the system. Put differently, one can always buy additional bandwidth but latency gains are often limited by the laws of physics. For instance, consider TCP's performance. Initially TCP is limited to sending only a few segments in the first RTT of data transmission. The data rate increases exponentially afterwards. However, over a 1 Gbps network path with a 50 msec RTT a TCP connection will require 10 RTTs and over 14 MB of data before utilizing the available capacity. Most transfers carry nowhere near enough data to achieve these speeds [4]. This shows that realizing high speed transfer is not as easy as simply adding raw capacity.

In addition to leveraging numerous content replicas as we sketch in § IV-B, we envision a more radical notion: keeping a local copy of the entire Internet. Instead of retrieving content on-demand over the wide-area network, users will access a local copy cached in the HPoP. This arrangement allows us to use the copious bandwidth within ultrabroadband networks to lower the users' perceived delay. Of course, even with the high capacity of ultrabroadband, copying the entire Internet is ridiculous! Therefore, a key task is in approximating an exact copy of the Internet for the given residence, which entails several facets, as follows.

Aggressiveness: Traditional prefetching of content tries to leverage users' actions to anticipate their near-term needs. Our approach is less fine-grain. We aim to leverage users' long-term history to copy the portion of the Internet the users visit and are likely to visit. This starts with understanding the requirements imposed by various snapshots of history, including storage requirements, core network workload and server workload. Given the HPoP's vantage point it can measure these aspects as part of the system's operation. These measurements can drive decisions, such as determining whether to keep content fresh by fetching a new copy as a cached version expires. Based on the effect on the upstream loads, the HPoP can evaluate the tradeoff between the extent of content gathering and the degree of its freshness. That is, we can decrease the number of requests going to the Internet by either reducing the scope of the content gathered (thus reducing the volume of requests necessary to keep the content fresh) or by decreasing the frequency of content pre-validation.

Deep Web Content: Dynamic and personal content—the so called "deep web" [38], [39]—now plays a large role on

the Internet. Therefore, to truly copy a suitable version of the Internet, the HPoP will hold user credentials so it can copy deep web content, e.g., constantly collect comments on user's Facebook page to make them locally available whenever needed, or content from websites that require subscription. While divulging credentials for web mail or social networking services to some generic web proxy would be unthinkable, providing these to a device in a user's own house and ultimately under their control is much more palatable.

We note that some Internet applications already implement certain aspects of automatic client-side interactions, such as the Calibre system for downloading news feeds and repackaging them into an e-book [40]. HPoP's deep web content gathering will enrich these functionalities and support them in a generic fashion across sites.

Leveraging the Data Attic: The Internet@home and data attic aspects of HPoP can work synergistically, in that data attic provides a rich source of information for HPoP's decisions on the portions of the Internet to maintain locally. For instance, by gathering stock ticker symbols from tax documents the HPoP can maintain fresh stock quotes that are germane to the users. The HPoP will provide a generic modular framework such that many forms of information within the data attic can trigger data collection. As the amount of information users concentrate in their data attic increases, its value to this decision making grows.

Demand Smoothing: While aggressive content copying increases the load upstream, it can also mitigate potential bottlenecks upstream. Recall that by removing the last-mile floodgates, FTTH may strain Internet servers and core networks, and aggressive content copying as sketched above can exacerbate this effect. However, obtaining content ahead of actual use also brings flexibility to schedule content acquisition at an opportune time. This can smooth the demand on Internet servers and core networks.

A Cooperative Cache: As sketched in § II one of the aspects of FTTH is that other edge networks in topological proximity will also have a wealth of capacity while the aggregate connection from the neighborhood may be a bottleneck. Therefore, neighboring HPoPs can link together to coordinate their content gathering activities and avoid duplicate retrievals and storage of content in an effort to save aggregate capacity to the neighborhood. Content can then be shared by all hosts within the community in a peer-to-peer manner. This is a resurrection of cooperative Web caching popular in the late 1990s (see, e.g., [41] and reference therein), but extended with coordinated content gathering.

V. Related Work

We are not aware of prior work on the notion of centering users' digital lives around an appliance—such as the HPoP—in their homes. In terms of the specific HPoP services we plan to explore, our data attics address mainly the issue of data control in the clouds and external applications. While the problem has been well recognized [12], [42], [43], most previous work addressing this issue focuses on data security through cryptography [44], [45], auditing and accountability [46], [47], access control [48], and security of the cloud platform itself [43]. These approaches certainly raise the bar for an attacker but leave the user tied to a given cloud provider and unable to directly protect the data from inappropriate use or accidental leak once the cloud's security is breached. An integral component of data attics is peer backup, which has seen much prior research (e.g., [49]–[53]). Data attics can benefit from many of these solutions.

The NoCDN service takes inspiration from peer-to-peer distribution of streaming media (see the survey [54] and references therein) and file sharing such as BitTorrent. Unlike these approaches, NoCDN functions like a traditional CDN with residential peers replacing the CDN servers. This arrangement has the potential of creating a new marketplace where any web site can recruit peers to help with their content delivery using their own incentives. While a large amount literature examines the possibility of incorporating residential peers into a traditional CDN, we are exploring a possibility of removing the CDN altogether for some content providers. Our technique for ensuring data integrity is similar to Stickler [55], although Stickler requires content providers to drastically change content organization, by replacing regular embedded objects with recursively nested JavaScript scripts. NoCDN only adds wrapper meta-pages, leaving the actual content intact.

Our Internet@home service draws on Web prefetching (with related work too numerous to list) and deep web crawling (e.g., [38], [39]). While leveraging existing technologies to the extent possible, our new context and motivation will inevitably require new techniques.

VI. Conclusion

Our over-arching thesis is that ultrabroadband opens am opportunity to re-think and re-organize how networks, applications and information are handled in the Internet. We propose a general direction for this re-thinking: the significant jump in home network capacity warrants a corresponding shift in the responsibility assumed by the home network in supporting users' Internet activities. Our model calls for assigning this responsibility to a "home point of presence" (HPoP). The HPoP is meant to be a hub around which much of users' digital lives can revolve. While we discuss a number of promising services the HPoP can provide, our intention is that the HPoP be a generic device that can evolve and accrue further responsibilities over time. Finally, note that while we have adopted an approach based on an HPoP, we hope the community will embrace the high-order question of how to better use ultrabroadband residential networks.

References

[1] "Faster, Fairer, Kinder Internet," https://fiber.google.com/about/.

[2] "Your Gig is Here," http://chattanoogagig.com/, 2010.

[3] "World's First Community-wide 10 Gigabit Internet," http://www.futuretimeline.net/blog/2015/10/17.htm, 2015.

[4] M. Sargent and M. Allman, "Performance Within A Fiber-To-The-Home Network," *ACM Computer Comm. Review*, vol. 44, no. 3, Jul. 2014.

[5] "Case Connection Zone," http://caseconnectionzone.org/.

[6] M. Boucadair, R. Penno, and D. Wing, "Universal Plug And Play (UPnP) Internet Gateway Device - Port Control Protocol Interworking Function (IGD-PCP IWF)," RFC 6970, Jul. 2013.

[7] J. Rosenberg, J. Weinberger, C. Huitema, and R. Mahy, "STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs)," RFC 3489, Mar. 2003.

[8] R. Mahy, P. Matthews, and J. Rosenberg, "Traversal Using Relays Around NAT (TURN): Relay Extensions To Session Traversal Utilities For NAT (STUN)," RFC 5766, Apr. 2010.

[9] "State of IPv6 Deployment 2018. Internet Society." https://www.internetsociety.org/wp-content/uploads/2018/06/2018-ISOC-Report-IPv6-Deployment.pdf.

[10] J. Xu and M. Rabinovich, "NoCDN: Scalable Content Delivery Without a Middleman," in *Proc. of the 5th ACM/IEEE HotWeb Workshop*, 2017.

[11] S. Brennan and M. Rabinovich, "Improving Communication through Overlay Detours: Pipe Dream or Actionable Insight?" in *Proc. of the IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, 2018, pp. 1422–1431.

[12] C. Soghoian, "Caught in the Cloud: Privacy, Encryption, and Government Back Doors in the Web 2.0 Era," *Journal on Telecommunications and High Technology Law*, vol. 8, no. 2, 2010.

[13] R. Geambasu, T. Kohno, A. Levy, and H. Levy, "Vanish: Increasing Data Privacy With Self-Destructing Data," in *Proc. of the 18th USENIX Security Symposium*, 2009, pp. 299–316.

[14] "About Google Docs Offline," http://support.google.com/docs/bin/answer.py?hl=en&answer=1628467.

[15] J. Kistler and M. Satyanarayanan, "Disconnected Operation in the Coda File System," *ACM Trans. on Computer Systems (TOCS)*, vol. 10, no. 1, pp. 3–25, 1992.

[16] L. Huston and P. Honeyman, "Disconnected Operation for AFS," in *Proc. of the USENIX Symposium on Mobile and Location-Independent Computing*, 1993, pp. 1–10.

[17] R. Gruber, F. Kaashoek, B. Liskov, and L. Shrira, "Disconnected operation in the Thor object-oriented database system," in *Proc. of the 1st IEEE Workshop on Mobile Computing Systems and Applications*, 1994, pp. 51–56.

[18] http://www.microsoft.com/en-us/healthvault/.

[19] http://www.medefile.com/.

[20] L. Rizzo, "Effective Erasure Codes for Reliable Computer Communication Protocols," *ACM Computer Comm. Review*, vol. 27, no. 2, pp. 24–36, 1997.

[21] H. Yin, X. Liu, T. Zhan, V. Sekar, F. Qiu, C. Lin, H. Zhang, and B. Li, "Livesky: Enhancing CDN with P2P," *ACM Trans. Multimedia Comput. Commun. Appl. (TOMM)*, vol. 6, no. 3, pp. 1–19, 2010.

[22] M. Zhao, P. Aditya, A. Chen, Y. Lin, A. Haeberlen, P. Druschel, B. Maggs, B. Wishon, and M. Ponec, "Peer-assisted content distribution in Akamai NetSession," in *Proc. of the ACM Internet Measurement Conference*, 2013, pp. 31–42.

[23] S. Hanselman, "CSI: My Computer - What is netsession_win.exe from Akamai and How Did It Get On My System?" Available at goo.gl/wgM1CK (Accessed on 08/03/2017), 2011.

[24] P. Rodriguez, A. Kirpal, and E. W. Biersack, "Parallel-Access for Mirror Sites in the Internet," in *Proc. of the IEEE INFOCOM*, 2000.

[25] A. Barbir, B. Cain, R. Nair, and O. Spatscheck, "Known Content Network (CN) Request-Routing Mechanisms," RFC 3568, Jul. 2003.

[26] S. Savage, T. Anderson, A. Aggarwal, D. Becker, N. Cardwell, A. Collins, E. Hoffman, J. Snell, A. Vahdat, G. Voelker, and J. Zahorian, "Detour: Informed Internet routing and transport," *IEEE Micro*, vol. 19, no. 1, pp. 50–59, 1999.

[27] P. K. Gummadi, H. V. Madhyastha, S. D. Gribble, H. M. Levy, and D. Wetherall, "Improving the Reliability of Internet Paths with One-hop Source Routing," in *Proc. of Usenix Symposium on Networked Systems Design and Implementation (NSDI)*, 2004.

[28] H. Zheng, E. K. Lua, M. Pias, and T. G. Griffin, "Internet Routing Policies and Round-Trip-Times," in *Proc. of the Passive and Active Measurement Conf. (PAM)*, 2005, pp. 236–250.

[29] S.-J. Lee, S. Banerjee, P. Sharma, P. Yalagandula, and S. Basu, "Bandwidth-Aware Routing in Overlay Networks," in *Proc. of IEEE INFOCOM*, 2008, pp. 1732–1740.

[30] C. Lumezanu, R. Baden, D. Levin, N. Spring, and B. Bhattacharjee, "Symbiotic Relationships in Internet Routing Overlays," in *Proc. of the USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2009, pp. 467–480.

[31] Akamai, "SureRoute," https://developer.akamai.com/learn/Optimization/SureRoute.html.

[32] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris, "Resilient Overlay Networks," in *Proc. of the 18th ACM Symposium on Operating Systems Principles (SOSP)*, 2001, pp. 131–145.

[33] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure, "TCP Extensions For Multipath Operation With Multiple Addresses," RFC 6824, Jan. 2013.

[34] O. Mehani, R. Holz, S. Ferlin, and R. Boreli, "An Early Look at Multipath TCP Deployment in the Wild," in *Proc. of the 6th International Workshop on Hot Topics in Planet-Scale Measurement*, 2015, pp. 7–12.

[35] O. Bonaventure and S. Seo, "Multipath TCP deployments," *IETF Journal*, vol. 12, no. 2, pp. 24–27, 2016.

[36] M. Boucadair, C. Jacquenet, O. Bonaventure, D. Behaghel, S. Secci, W. Henderickx, R. Skog, S. Vinapamula, S. Seo, W. Cloetens, U. Meyer, L. M. Contreras, and B. Peirens, "Extensions for Network-Assisted MPTCP Deployment Models," 2017, IETF Draft. [Online]. Available: https://tools.ietf.org/html/draft-boucadair-mptcp-plain-mode-10

[37] A. P. Felt, R. Barnes, A. King, C. Palmer, C. Bentzel, and P. Tabriz, "Measuring HTTPS Adoption on the Web," in *Proc. of the 26th USENIX Security Symposium*, 2017, pp. 1323–1338.

[38] B. T OGRAPH, Y. AMANCA, and Y. MAAHS, "Searching the Deep Web," *Communications of the ACM*, vol. 51, no. 10, 2008.

[39] J. Madhavan, D. Ko, Ł. Kot, V. Ganapathy, A. Rasmussen, and A. Halevy, "Google's Deep Web Crawl," *Proc. of the VLDB Endowment*, vol. 1, no. 2, pp. 1241–1252, 2008.

[40] "calibre User Manual," http://manual.calibre-ebook.com/.

[41] M. Rabinovich and O. Spatscheck, *Web Caching and Replication*. Addison-Wesley, 2001.

[42] L. Kaufman, "Data Security in the World of Cloud Computing," *IEEE Security & Privacy*, vol. 7, no. 4, pp. 61–64, 2009.

[43] B. Kandukuri, V. Paturi, and A. Rakshit, "Cloud Security Issues," in *IEEE Int. Conf. on Services Computing (SCC)*, 2009, pp. 517–520.

[44] S. Kamara and K. Lauter, "Cryptographic Cloud Storage," in *Proc. of the Int. Conf. on Financial Cryptography and Data Security*. Springer, 2010, pp. 136–149.

[45] R. Chow, P. Golle, M. Jakobsson, E. Shi, J. Staddon, R. Masuoka, and J. Molina, "Controlling Data in the Cloud: Outsourcing Computation Without Outsourcing Control," in *Proc. of the ACM Workshop on Cloud Computing Security*, ser. CCSW '09, 2009, pp. 85–90.

[46] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing," in *Proc. of IEEE INFOCOM*, 2010.

[47] S. Pearson and A. Charlesworth, "Accountability as a Way Forward for Privacy Protection in the Cloud," *Cloud Computing*, pp. 131–144, 2009.

[48] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving Secure, Scalable, and Fine-Grained Data Access Control in Cloud Computing," in *Proc. of IEEE INFOCOM*, 2010.

[49] A. Haeberlen, A. Mislove, and P. Druschel, "Glacier: Highly Durable, Decentralized Storage Despite Massive Correlated Failures," in *Proc. of the USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2005.

[50] A. Duminuco and E. Biersack, "A Practical Study of Regenerating Codes for Peer-to-Peer Backup Systems," in *Proc. of the 29th IEEE Int. Conf. on Distributed Computing Systems (ICDCS)*, 2009, pp. 376–384.

[51] M. Lillibridge, S. Elnikety, A. Birrell, M. Burrows, and M. Isard, "A Cooperative Internet Backup Scheme," in *Proc. of the USENIX Annual Technical Conference*, 2003.

[52] L. P. Cox, C. D. Murray, and B. D. Noble, "Pastiche: Making Backup Cheap and Easy," in *Proc. of USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2002.

[53] A. Adya, W. J. Bolosky, M. Castro, G. Cermak, R. Chaiken, J. R. Douceur, J. Howell, J. R. Lorch, M. Theimer, and R. Wattenhofer, "FARSITE: Federated, Available, and Reliable Storage for an Incompletely Trusted Environment," in *Proc. of USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2002.

[54] S. Androutsellis-Theotokis and D. Spinellis, "A Survey of Peer-to-Peer Content Distribution Technologies," *ACM Comput. Surv.*, vol. 36, pp. 335–371, December 2004.

[55] A. Levy, H. Corrigan-Gibbs, and D. Boneh, "Stickler: Defending Against Malicious Content Distribution Networks in an Unmodified Browser," *IEEE Security & Privacy*, vol. 14, no. 2, pp. 22–28, 2016.