The CHIANTI Architecture for Robust Mobile Internet Access

Jörg Ott
Petri Ylikoski
TKK Comnet

Nils Seifert
LYSATIQ GmbH

Caleb Carroll
Nigel Wallbridge
NOMAD Digital Ltd.

Olaf Bergmann
Dirk Kutscher
Uni Bremen TZI

Abstract

The quality of Internet access for mobile users may suffer from highly variable communication characteristics (packet loss, delay, throughput) and from temporary disconnections. The former occur as a result of changing radio properties, handovers, or variable system load, the latter whenever wireless coverage is insufficient. While quite some research has tackled improving radio coverage to keep users always best connected and numerous approaches pursue improving wireless performance and robustness, these often assume a greenfield deployment or a tight integration with operators. We present the mobility support system architecture developed in the CHIANTI project. While building on related work in various technical respects, the CHIANTI architecture is specifically designed to be instantly and incrementally deployable in today's Internet landscape, considering real-world legal and deployment constraints, and supports roles of different (independent) players. We also report on our proof-of-concept implementation.

1. Introduction

Mobile Internet access has become a commodity in recent years, with broad availability of (commercial and free) WLAN hot-spots, more appealing (flat rate) high-speed cellular data subscriptions, and, increasingly with WLAN-based infrastructure in vehicles such as trains and buses. While wireless access technologies offer higher data rates and wireless network coverage continuously increases, we are still far from seamless and ubiquitous connectivity. Particularly when users move, wireless conditions change and handovers (potentially to different operators or networks) occur, that may yield significantly different communication characteristics—or connectivity may be lost altogether. Even though Internet protocols and applications are (or should be) designed to adapt to changing network conditions, their capabilities to deal with sudden changes are fairly limited—and disconnections are usually not tolerated at all. The former may and the latter usually do lead to errors, reported to the user for manual repair/retry [25].

Mobility support solutions were developed for many layers (e.g., [32, 6, 9, 21, 23, 18, 38]), performance enhancement mechanisms were designed for various challenged link layers such as satellite and wireless communications (e.g., [36, 1, 8, 31, 14, 2]), and various disconnection tolerance mechanisms were suggested (see section 2). And system solutions for mobility in cellular networks and across network boundaries were designed, most prominently by 3GPP. What is missing, however, is a comprehensive architecture that takes into account real-world constraints *and* does not depend on a coherent operator deployment, thus being open to incremental introduction into the market by independent players.

The CHIANTI project is pursuing the development, implementation, and initial deployment of such an infrastructure in operational networks supporting wireless Internet access from trains. In this paper, after reviewing related work specifically on disconnection tolerance in section 2, we outline the CHIANTI reference scenario in section 3 and discuss constraints arising from today's real-world environment in section 4. We describe the CHIANTI system architecture in section 5, highlighting those aspects relevant for immediate and incremental deployment, and report on our initial implementation 6. We conclude this paper with a brief summary and hint at future work in section 7.

2. Related Work

The literature on node and network mobility and performance enhancements for wireless communications is vast and so we focus our review on related work on disconnection tolerance. While some end-to-end approaches were developed to enable (mostly TCP-based) applications to better deal with disconnections (discussed first), most approaches pursue some variant of connection splitting, i.e., introducing one or more intermediaries that terminate the transport/application connections on both sides of a challenged link and run an enhanced protocol between those intermediaries. While this allows applications to remain unchanged, it introduces dependencies on additional elements.

978-1-4244-4439-7/09/$25.00

FreezeTCP [16] operates end-to-end to pause and resume TCP communications to avoid timeouts. This feature was combined with outage prediction for vehicle-based communications (buses running regular routes) [5]. TCP Migrate [37] modifies TCP for end-to-end disconnection tolerance. It uses dynamic DNS to cope with IP address changes and new TCP option mechanisms to migrate TCP connections across address changes creating the illusion of connection persistence. Its operation requires advance knowledge of impending address changes which may not be available in network environments with unpredictable disconnections (and thus lead to disruptions). Based upon TCP Migrate, the Transparent, Extensible Session-Layer Architecture (TESLA) [33] was developed as an end-to-end framework to offer different session layer services for networked applications. It is implemented as a library to provide a transparent shim layer between application and transport layers and thus also requires modifications at both ends. And for SSH, an example of a TCP-based application protocol, enhancements were developed to support suspending/resuming communication relationships [19].

Indirect TCP (I-TCP, [6]) uses just one intermediary but requires modifications to the mobile host. It is designed for mobile Internet access over a single-hop wireless link. The path is split at the base station of the wireless network. I-TCP supports handovers between multiple base stations for moving mobile hosts. Temporary disconnections are not visible to the peer host in the fixed network.

Two intermediaries in the path are used by the FleetNet [7] architecture that focused on ad-hoc networking between cars to reach roadside access points for vehicular Internet access, employing connection splitting by means of proxies in the vehicle and in the access points. DHARMA [22] is an overlay network utilizing distributed proxy-like home agents for disconnection tolerance and location-awareness for legacy TCP applications. While DHARMA supports multiple wireless interfaces, its restriction to TCP limits its general applicability. The Drive-thru Internet architecture [26, 27] with its Persistent Connection management Protocol (PCMP) [28] offers session management between proxy servers. It provides persistent connections over and resilience to changing IP addresses while maintaining transparency to legacy applications. While its design is explicitly targeted at supporting different transport protocols, including UDP, the current implementation only covers TCP. The Opportunistic Connection Management Protocol (OCMP) [35] builds upon some of the concepts of PCMP, developing them further to also offer proxy failover and some policy-based use of multiple wireless interfaces. OCMP also supports other transport protocols and, like PCMP, offers the possibility to include application-specific plug-ins.

While the design goals of FleetNet, DHARMA, PCMP, and OCMP are somewhat similar to those of CHIANTI, comprehensive considerations of market and deployment constraints have not received much attention before, with two exceptions: The Drive-thru Internet project [26] that uses PCMP explicitly discussed the issue of opportunistic access to (commercial) WLAN hot-spots and the need for portal-based authentication [29]; and the KioskNet project [17] that uses OCMP has demonstrated real-world deployment for Internet kiosks in India. KioskNet has a different target scenario and can thus pursue a greenfield deployment. The CHIANTI architecture builds on the Drive-thru Internet concepts and on the authors' experience in designing performance enhancement systems for satellite communications.

Finally, delay-tolerant networking [15] addresses communication in challenged networks, with long latencies, high loss rates, and partial or temporary connectivity, where end-to-end connectivity may never be available. Typical DTN scenarios include communications in deep space, military units in combat, simple sensor networks, ad-hoc networks in disaster areas. One DTN architecture [10] consists of a message-based overlay network using a store-and-forward mechanism with large messages (DTN bundles). The CHIANTI scenario forms a special case in the DTN space, yet DTN technologies have only limited applicability for CHIANTI, DTN requires application protocols designed to operate based on asynchronous communications which is not the case with legacy Internet transport and application protocols. Yet, DTN protocols are investigated as one communication option for the CHIANTI architecture.

3. Scenarios and Reference Model

We target two different mobility scenarios: *Nomadic users* are stationary in a fixed location while using a wireless access network (e.g., in a café) and have their applications suspended while moving between locations. Thus, they (usually) exercise controlled connection and disconnection to/from the wireless network and their environmental conditions are expected to change less (at least due to their own motion). *Mobile users* run their applications while moving and those are hence subject to uncontrolled and often unpredictable changes in connectivity. Mobile users, e.g., on a train, often also exhibit nomadic behavior when turning off their devices while entering and leaving.

Figure 1 depicts a typical high-level network setup for mobile Internet access from trains: a WLAN on the train offers connectivity to mobile users via one or more different access networks. These access networks connecting the trains to the Internet may (left side) or may not (right side) be operated by a CHIANTI ISP, supporting disconnection tolerance and other adaptation mechanisms. In practice, mixed scenarios may prevail, as is the case for Nomad as a service provider in the UK: the company has rolled out trackside wireless access networks based upon WiMAX

technologies but also uses packet data access via GSM and UMTS networks from mobile operators. Nomad runs an overlay derived from mobile IP to support mobile access across heterogeneous networks, with one tunnel endpoint in a mobile router on the train and the other in their network infrastructure. CHIANTI functionality will add to this setup (see section 5) by introducing a minimal overlay of CHIANTI nodes for dealing with connectivity changes.

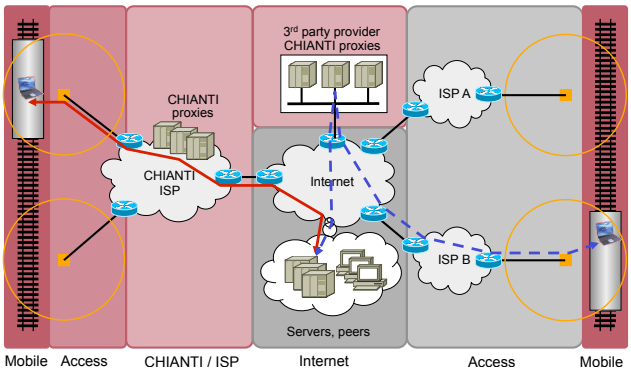

Figure 1. Reference Model and Spheres

3.1 CHIANTI Spheres

We use the term *sphere* to denote a topological region of a network under control of a single entity or a group of entities. The term has been chosen to facilitate delimiting the administrative or business boundaries between the involved players. Figure 1 indicates the different spheres at the bottom and shows their boundaries.

The *Internet sphere* (grey-shaded in the middle in figure 1) refers to unchanged IP-based transit networks without any CHIANTI-specific functionality. This sphere hosts the majority of servers and peers of interest to CHIANTI users, be they general web- or email servers, access gateways to private corporate of home networks or peers.

The *Access sphere* provides Internet access for mobile users or vehicles. The (CHIANTI) Access sphere (red-shaded on the left in figure 1) represents an access service provider which has support for CHIANTI-related functions, indicated by the presence of CHIANTI proxies within the ISP sphere hosting the access infrastructure. In contrast, the grey-shaded access sphere on the right of the figure shows an access service provider without any such support. In the latter case, CHIANTI functions have to be provided deeper inside the network by third party providers (see below), which causes indirections and thus may lead to suboptimal routing and increased latency. Access spheres without CHIANTI support may contain other servers and infrastructure related to ISP functions, such as firewalls, email servers, portals, NATs, etc.

The *(CHIANTI) Service Provider sphere* denotes the sphere which contains Internet service providers with CHIANTI support functions, i.e., a service provider's CHIANTI nodes. This sphere may contain other ISP-related infrastructure, such as access portals, firewalls, filters, NAT servers, etc. CHIANTI services may also be provided by third party CHIANTI service providers who do not have their own access infrastructure, but instead host CHIANTI proxies somewhere further away from access networks. They may offer CHIANTI services to the general public without providing other ISP functionality and rely on other ISPs for connectivity (similar to other application service providers). In such a case, as shown in the figure above, the CHIANTI nodes are logically located in the Internet sphere (red-shaded, in the middle at the top). Thus, even enterprises or end users themselves could run CHIANTI proxies in their corporate, hosted, or home networks, e.g. to provide persistent service access.

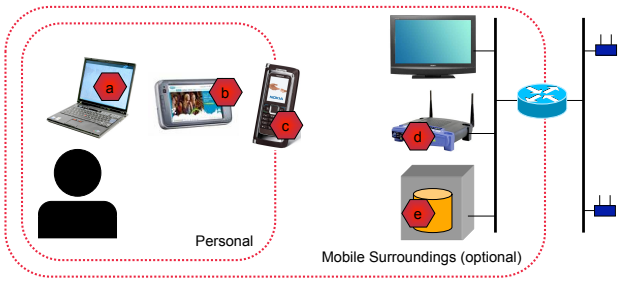

Figure 2. Mobile Sphere

Finally, the *Mobile Sphere* (on the extreme left and right in figure 1) refers to the mobile user and her immediate surroundings. She makes use of CHIANTI services to obtain robust and disconnection-tolerant access to Internet services. The mobile sphere may be considered to have two constituents as shown in figure 2: The *Personal sphere* (the inner box in the figure) includes all mobile devices belonging to a particular user, which are assumed to be well-known and trustworthy to each other. CHIANTI functionality realized on any of these devices (a, b, c in the figure) will hence be trustworthy, too, and always be available to the individual user (also in the nomadic use case).

The optional *Mobile Infrastructure* sphere (outer box in figure 2) includes all supportive communication devices available in the mobile environment: this may contain WLAN access points or mobile routers, WiMAX or 3G access modems for upstream connectivity, and possible proxy or content servers. And CHIANTI nodes, provided by the transportation company or some other service provider, may

be part of the mobile infrastructure (d, e), capable of serving all users traveling in the vehicle. Because of its shared nature and being run by a third party, this CHIANTI functionality may not be as trusted by individual users.

3.2 Network Performance & Applications

Mobile Internet access in the above settings may suffer from two major issues to be addressed by the CHIANTI service in order to offer a satisfactory user experience:

1) Data communication over cellular networks quite often sees high round-trip times, primarily due to significant queuing presumably introduced by (access) routers at the edges of the core network, but also due to the wireless access technologies in use. The RTT on GRPS/EDGE networks easily grows to several seconds and, even in UMTS networks with HSDPA, several hundred milliseconds may be observed. Our measurements confirmed what has been repeatedly reported in other studies: We found some 100ms mean (500ms max) RTT for unloaded stationary HSDPA and 400ms mean for GPRS, going up to 1.1s (HSDPA) and 10+s (GRPS) mean RTT with a single TCP connection creating background traffic [11].

2) Network coverage may be partly poor or incomplete leading to connectivity disruptions as discussed above. We looked at the coverage for individual users using cellular networks in cars and trains along different roads and tracks as well as the connectivity available on various Nomad train deployments. While the former showed repeated disconnections, the latter featured more extensive coverage due to combining multiple access technologies (up to four); yet, there were still a few gaps to be filled by CHIANTI. Moreover, the available network capacity changed dramatically when switching between access technologies (WiMAX to HSPDA to GPRS), yielding the—expected—orders of magnitude difference in RTT and throughput [12].

High RTTs, as is well-known from satellite communications, seriously impede transport (TCP)[1] and application protocol performance; the latter particularly holds for applications that perform continuous interactions with their peers, such as web browsers iteratively retrieving objects of a web page. Experiments with retrieving six web pages showed a mean completion time of some 350ms for wired access and 1.5s for HSDPA without load, rising to 600ms and 100+s under load, respectively [11, 12]. Furthermore, applications that are not able to complete their transactions (e.g., downloading or sending an email) while connected, may have to start all over again when connectivity comes back, often requiring user involvement [24].

A preliminary study on an operational Nomad train service in Canada [11] revealed that nearly all users use TCP and a bit more than half UDP; 71% of all flows were HTTP and 11% HTTPS; DNS, gaming, and VPN traffic accounted for some 1–3%. Notable was the absence of (secure) SMTP and POP3/IMAP4; we attribute this to people using primarily web-based mail access and, to a lesser extent, corporate mail clients hidden inside VPNs. The key findings are that proper support for efficient and disconnection-tolerant TCP will cover a broad user basis, to be complemented by support for selected UDP protocols.

[1] TCP does not deal well with abruptly changing RTT (due to its RTT calculation) and path capacity (due to its congestion control mechanisms).

4. Real-World Constraints

Today's operational networks incorporate numerous pieces of network equipment which are, in principle, supposed to be invisible to end-to-end application protocols run between the communicating end points but, in practice, make certain assumptions about which applications are run and how the respective application protocols operate. We refer to these "intermediaries" as *third-party interferers* and discuss the architectural implications of their presence.

Figure 3 depicts a simplified network topology: the mobile user on the left is connected to a vehicle network with a CHIANTI node (C) that talks to its infrastructure peer (P). Different types of interferers (described below) may be found at different positions (1–4) in the path from the mobile device to the CHIANTI infrastructure peer or the Internet. For nomadic users, the CHIANTI functionality (C) would also reside inside the mobile device and the vehicle network would be, e.g., the WLAN hotspot.

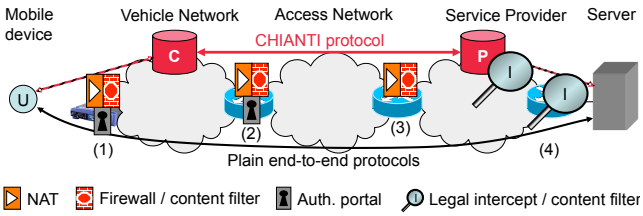

Figure 3. Network Topology and Interferers

4.1 Firewalls and NATs

Firewalls implement a variety of techniques to prevent (seemingly) unauthorized packets from passing. These include (but are not limited to):

Packet filtering: incoming/outgoing packets are matched against predefined quintuples (source and destination IP address and port number, protocol identifier), some of which may be wildcarded to determine whether or not a packet is allowed to pass. Responses to previously outgoing packets (e.g., for DNS queries or HTTP requests) are allowed.

Deep packet inspection: beyond simply looking at IP addresses and port number, the transport and application layer contents may also be inspected and possibly state created and maintained. In simple cases, TCP packets are checked against previously passed SYN/ACK handshakes. In more complex cases, the content of TCP connections on port 80 is checked to contain valid HTTP messages.

Application layer gatewaying: firewalls may also terminate a connection and then act as an application layer proxy or relay, in-depth inspecting the content of the connection also across several packets. This may be paired with content filtering and/or insertion as discussed in section 4.3.

NATs isolate two addressing realms and gateway between them. Originally intended to connect a "private" network to the global Internet, more complex scenarios with nested NATs have arisen over time. NATs map an (IP address, port number) pair inside the "NATed" network (e.g., of a user's mobile device) to another (IP address, port number) pair towards the outside [4]. A mapping is usually dynamically created upon a first outbound packet and maintained as long as traffic flows or whenever an end of flow is observed (e.g., from a TCP SYN-ACK to the FIN-ACK handshake. What is important is that timeouts for such address bindings of NATs vary and cannot be predicted and may even be applied to TCP connections. While (static) mappings may also be created for inbound packets, this is not feasible with mobile nodes and dynamically assigned addresses, so that NATs de-facto enforce unidirectional initiation of communication from the "inside".

NATs also apply firewall filtering functions: They match incoming and outgoing traffic according to the aforementioned binding and (optionally) additional filtering rules. Outbound matching simply ensures that packets to the same destination use the same source address and port. Inbound matching may be more sophisticated. Depending on the NAT implementation characteristics, packets from different remote peers may or may not be passed through the NAT even if a binding exists: in extreme cases, only remote peers to which packets were sent before are allowed to reply.

The major implications are that all communications must be initiated from the mobile device, that any protocol between CHIANTI nodes must be based upon UDP or TCP (possibly both to deal with different NAT and firewall policies), that such a protocol needs to support keep-alive mechanisms for NAT and firewall state, and that CHIANTI nodes have to deal with changing node addresses and thus need other means for endpoint identification.

4.2 Authentication Portals

Internet Access Points (like WLAN hotspots or hotel Internet access) often use authentication portals that require users to authenticate to the service before being allowed access (e.g., asking for name, credit card details or for pre-registered users for some name, password combination).

Often these portals use some form of captive interception, e.g., redirecting initial HTTP requests to redirect the user to a dedicated login page [3]. Capturing connections is usually performed by intercepting HTTP requests, terminating the connection at a local gateway, and issuing a "302 Redirected" HTTP response which points to an HTTPS URI of the local authentication server. While DNS packets pass, all other packets are dropped until the respective client has been authenticated so that non-HTTP traffic is silently blocked (usually, no clues about the failure, e.g., via ICMP messages, are provided either).

The location of such a portal is crucial: if the portal is topologically between the user application and the CHIANTI node C, i.e., position (1) in figure 3 (as is the case for mobile users with vehicle support) the user continues as usual for authentication. If the portal is located between the two CHIANTI nodes, C and P, i.e. at position (2) or (3) in figure 3 (as is the typical case for nomadic users and for mobile users without vehicle support), the presence of a captive portal means that node C may not be able to reach node P before user authentication has succeeded. In order for user authentication to succeed, however, C must pass HTTP requests from the user's browser so that the user can complete the login form by filling in the respective credentials. After successful authentication, however, the CHIANTI function should become active and apply the CHIANTI enhancements to further requests from the user web browser. This requires the CHIANTI node to monitor connectivity to the proxy and determine when a captive portal prevents packets from flowing and adapt its behavior accordingly.[2]

4.3 Content Filtering and Legal Intercept

CHIANTI service providers may want or have to implement content filtering functionality (e.g. Websense-based filtering of adult content) as soon as they appear as an Internet service provider, particularly when they secure the information exchange between CHIANTI nodes so that no access service provider or ISP can offer this functionality. The main design implication is that the respective filtering functions need to be co-located with CHIANTI nodes in the network infrastructure and the Internet.

On the other hand, content filtering or content adaptation may also be applied by third party firewalls or other devices thus potentially interfering CHIANTI protocol operation. If, for example, a firewall requires CHIANTI to utilize port 80 and an HTTP-style protocol encapsulation to

[2]Past experiments have shown that, while basically possible, automating authentication with wireless access portals is not practical today in spite of attempted standardization due to the many (subtle) differences in access portals [30].

reach its proxy, a third party content filter might still remove or modify—accidentally—essential parts of the CHIANTI protocol and prevent it from functioning properly. The main impact is that such interferer behavior needs to be considered as another reason for potential failure—to which CHIANTI functionality has to be dynamically adjusted or entirely disabled in order to avoid partial functioning and unpredictable operation from an end user perspective.

CHIANTI service providers may also need to realize legal interception (e.g., for monitoring individual access to remote location or the contents of such exchanges) and, similarly, other operators might (have to) implement this. Therefore, when a CHIANTI service provider performs encryption between client and the proxy (thus preventing, e.g., the access operator from performing certain interception functions), the CHIANTI service provider may need to track which individual users issue which requests and may need to be able to capture the communicated contents. If the CHIANTI service provider does not perform these functions but a third party operator is required to, the CHIANTI protocol operation may need to be adjusted—e.g., by configuration—to ensure that the legal requirements can still be met. Interception between the CHIANTI proxy and the Internet is not affected in any case.

5. System Architecture

As noted above, the CHIANTI system architecture takes a connection splitting approach, borrowing from the Drive-thru Internet architecture [27]: One CHIANTI node, is placed in the mobile sphere (the *CHIANTI client*), installed as part of the vehicular infrastructure or integrated with the mobile users' devices. Its peer node (the *CHIANTI proxy*) is located in the fixed infrastructure, at an ISP, a third-party service provider, an enterprise, or a user's home. The system architecture does not mandate any specific location. Figure 1 shows CHIANTI proxies as part of an operator network and as a third-party service and figure 3 the simple case of two interacting CHIANTI nodes.

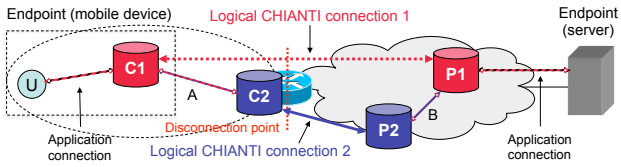

Figure 4. Nesting of CHIANTI functions

More sophisticated cases may arise if a (nomadic) user has a CHIANTI client installed on her device and then enters a train with CHIANTI support as part of the vehicular infrastructure (figure 4). The CHIANTI system is kept strictly modular and such a case will lead to nested support, which is beneficial to the user: while she maintains control of her application state on her mobile device (and does not need to trust external nodes), the vehicular infrastructure may offer better wireless access capabilities than the user's phone or laptop and more effective and efficient disconnection management as the train systems might be aware of upcoming handovers and potential disconnections. We foresee a local announcement channel inside a vehicular infrastructure to inform user devices about the availability of CHIANTI clients and enable users to choose which services to use [13]. Similarly, concatenation of CHIANTI functionality may occur if two users with CHIANTI support run peer-to-peer protocols directly between their clients. Nesting and concatenation may be combined.

In any of these scenarios, clients and proxies always operate as pairs. Clients determine their peer proxy via DNS (which could also serve load balancing using SRV records) and stay with the same proxy unless they terminate their application state or stay disconnected for too long (currently, we use a timeout of one week), in which case the server discards their state. A more sophisticated structure could be built upon this (e.g., by realizing failover mechanisms and state sharing among proxies similar to OCMP), but more complex overlay routing is not foreseen at this point to keep the provider infrastructure simple and manageable.

5.1 Functional Modules

Figure 5 depicts the functionality realized in the CHIANTI client and proxy nodes. The core communication functions comprise the upper three modules in both nodes. The *CHIANTI communication* function realizes the basic CHIANTI communication protocol between the client and the proxy. It provides a basic framing and offers data and control channels for information exchange for the other functions. *Local connection termination and handling (client)* is responsible for accepting incoming (TCP) connections and/or (UDP) packets from applications running on the mobile device(s) and dispatches them to the appropriate CHIANTI client modules for further handling. *Remote connection initiation (proxy)* realizes outbound (TCP) connections and/or outgoing (UDP) packets from the CHIANTI proxy to end systems in the Internet. This activity may be triggered by the client or may be based upon own decisions of the proxy. The *Persistent connection support (and optional enhancement)* function realizes disconnection tolerance for end-to-end connections and thus is the common building block for shielding applications from disruptions.

On top of disconnection tolerance and network adaptation support, *application-specific functions* may be added to implement selected aspects of individual application protocols to further improve the robustness to disruptions and

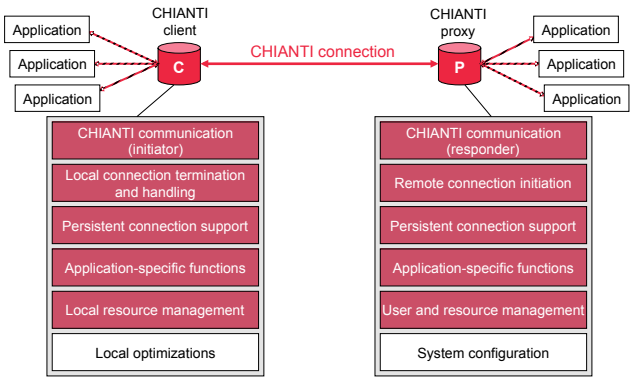

Figure 5. Client and Proxy Functionality

the overall application performance. Due to the dominance of HTTP, some minimal HTTP support is foreseen (recognizing that a full HTTP prefetching solution would be well beyond the project's scope and resources). In order to address CCTV applications used by the railroad companies, (RTSP-based) media streaming support is being developed.

Since CHIANTI clients and proxies may serve multiple users and different applications, resource management functions are provided to arbitrate the use of local and communication resources. The *local resource management (client)* maintains memory, CPU, bandwidth and other resources and ensures their arbitration across the local users according to predefined policies. The *user and resource management (proxy)* oversees the resource utilization by many clients connecting to a single proxy and arbitrates memory, CPU, bandwidth, and other resources according to predefined policies and/or current/past utilization.

The *system configuration (and coordination and monitoring)* function offers the overall configuration of the CHIANTI proxy (and polices its interaction with CHIANTI clients). It may, e.g., provide an interface to a user management data base, billing functions, etc. It might also offer some kind of management / monitoring functions to coordinate with other CHIANTI proxies of the same provider for load balancing, failover, and other related mechanisms.

Finally, *local optimizations* may cover local performance enhancements and interaction between CHIANTI clients. Such functions may include interaction with other entities in the local environment to learn about upcoming or imminent loss or gain of network connectivity (as could be announced by a wireless access module) and to provide/receive information about CHIANTI nodes in the vehicular infrastructure (and possibly their capabilities). Optimizations may also include sharing resources between several co-located mobile devices (e.g., caching).

5.2 Protocols and Interfaces

These functions interact with each other by means of system-internal interfaces (i.e., inside the client or proxy) and inter-system interface (i.e., between the client and proxy as well as between the CHIANTI nodes and end systems). Figure 6 shows the various external interfaces and the CHIANTI protocols (management and configuration interfaces and protocols are not depicted for clarity).

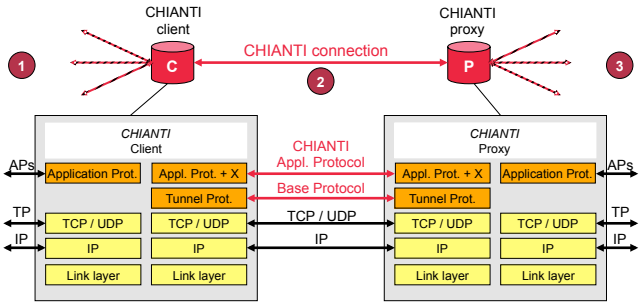

Figure 6. Protocols and Interfaces

The interfaces (1) and (3) are towards the (unmodified) applications running on the mobile devices and/or on hosts in the Internet. They use the regular Internet protocol suite at the IP and transport layer and standardized or proprietary application layer protocols. The interface itself operates at the transport layer and terminates packet forwarding (for UDP and optionally other datagram protocols) and transport connections (for TCP and optionally other connection-oriented protocols. This means that, in principle, every application protocol running on top of UDP or TCP could be enhanced if directed to a client interface (1). Application-specific protocol support is not a necessary prerequisite for offering the CHIANTI service to a particular application, at least for "short" disconnection periods. However, depending on the application protocol operation, additional application functions may be needed to deal with "longer" disconnections. "Short" and "long" are relative to a specific application, the respective application protocol and its configuration/parameterization, and a particular implementation and cannot be generalized.

Interface (2) is between the two CHIANTI nodes. Similar to interfaces (1) and (3), it uses IP and UDP or TCP as the basic communication protocols, so that NAT and firewall traversal will work. A CHIANT-specific framing and multiplexing protocol is provided on top—referred to as the *tunnel* protocol since application connections are ultimately tunneled through it. This protocol offers data and control channels. On top, besides a CHIANTI connection management between the CHIANTI client and proxy, application-specific protocols run natively or complemented by some

extensions as needed for disconnection tolerance and performance enhancement (referred to as "Appl. Prol. + X" in Figure 6 with "+ X" hinting at potential extensions). This standardized design at the IP and transport layers enables recursive enhancement of a CHIANTI connection of one service provider by a second pair of CHIANTI client and proxy operated by the same or a different service provider.

Two tunnel protocols are being implemented: 1) a simple UDP/TCP-based protocol developed based upon the authors' earlier experience with PCMP and performance enhancement protocols; 2) a protocol using the DTN bundle protocol over TCP/UDP [34]. The system design allows application protocols and potential extensions to be agnostic to the choice of the tunneling protocol. This flexibility facilitates experimentation and allows evolving the protocols over time based upon implementation and usage experience.

6. Implementation and Initial Validation

Figure 7 shows the structure of a CHIANTI node, termed *FlexProxy*, in this case the client side. The node uses, e.g., SOCKS [20] to intercept TCP connections or UDP flows from the application in the *interception module*). The application data may be subject to further processing (depicted as a *compression module*) and then handed off to the *DP-Basic module* that implements the simple variant of the tunnel protocol. Communication between CHIANTI nodes is managed by the *Inter Proxy module*. We use *connectors* and *junctions* as abstractions to link arbitrary modules inside the FlexProxy. In addition, external modules may be included—again using the SOCKS mechanism—by means of *external module adapters* (not shown). This allows for flexibility to extend the system, e.g., to include application-specific modules or try out other tunnel protocols, without requiring access to the source code or recompiling.

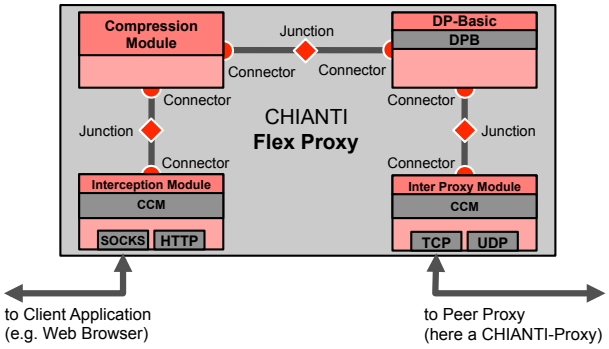

Figure 7. CHIANTI Node Architecture (client)

All the modules depicted in figure 7 are implemented (running on MS Windows, Linux, and MacOS X), with the simple tunnel protocol used for communication between CHIANTI nodes. We validated the basic functionality with two simple applications, SSH and HTTP, used when moving in and out of WLAN hotspots. An SSH session can be kept alive (including all tunneled connections) for up to the timeout (1 week) configured in the CHIANTI system, provided that the SSH connection does not use SSH layer keep-alives for maintaining NAT bindings. HTTP-based downloads of a large file (a Linux distribution) resumed whenever connectivity came back, web form submissions did not fail while disconnected but completed successfully after reconnection, and accessing web pages became more robust compared to not using CHIANTI. In all cases, resuming communication took less than 10s (without any link layer indications about network availability).

7. Conclusion

In this paper, we have introduced the CHIANTI architecture, a proxy-based system for supporting legacy applications in the presence of intermittent connectivity and changing connectivity characteristics. The system is targeted at individual use by mobile and nomadic users as well as at installation in vehicular infrastructure supporting groups of travelers. While building on a variety of (partly the authors' own) previous work, the CHIANTI approach focuses on including market requirements—incremental deployability, compatibility with today's operating environments, consideration of legal requirements, support for legacy applications, and applicability for different players. The software system design specifically considers extensibility and its initial prototype runs on the major operating systems.

While we have so far carried out only an initial validation of the overall concept and its implementation, the next steps include systematic performance assessment for the basic CHIANTI protocol and for CHIANTI over DTN as well as for the presently developed application-specific modules. Overall, the system has proven stable to serve as a daily platform for some of the project members to circumvent repeated SSH tunnel setup. This also demonstrates the value of incremental deployability of the CHIANTI system.

8. Acknowledgments

The authors would like to thank Kevin Loos for fruitful discussions. The research leading to these results has received funding from the European Community's Seventh Framework Programme under grant agreement no 216714.

References

[1] M. Allman, D. R. Glover, and L. A. Sanchez. Enhancing TCP Over Satellite Channels using Standard Mechanisms.

RFC 2488, January 1999.

[2] M. Allman, C. Hayes, H. Kruse, and S. Ostermann. TCP Performance Over Satellite Links. In *Proceedings of the 5th International Conference on Telecommunication Systems*, March 1997.

[3] B. Anton, B. Bullock, and J. Short. Best Current Practices for Wireless Internet Service Provider (WISP) Roaming, Version 1.0. Wi-Fi Alliance, February 2003.

[4] F. Audet and C. Jennings. Network Address Translation (NAT) Behavioral Requirements for Unicast UDP. RFC 4787, January 2007.

[5] A. Baig, M. Hassan, and L. Libman. Prediction-based Recovery from Link Outages in On-Board Mobile Communication Networks. In *Proceeding of IEEE Globecom 2004*, December 2004.

[6] A. Bakre and B. Badrinath. I-TCP: Indirect TCP for Mobile Hosts. Technical report, Department of Computer Science, Rutgers University, October 1994.

[7] M. Bechler, W. J. Franz, and L. Wolf. Mobile Internet Access in FleetNet. In *13. Fachtagung Kommunikation in verteilten Systemen, Leipzig, Germany*, April 2003.

[8] J. Border, M. Kojo, J. Griner, G. Montenegro, and Z. Shelby. Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations. RFC 3135, June 2001.

[9] K. Brown and S. Singh. M-TCP: TCP for Mobile Cellular Networks. *ACM Computer Communication Review*, 27(5), 1997.

[10] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H.Weiss. Delay-Tolerant Network Architecture. RFC 4838, April 2007.

[11] CHIANTI Project. Operational and User Requirements. Deliverable D1.2, June 2008.

[12] CHIANTI Project. Protocol analysis report. Deliverable D2.1, August 2008.

[13] CHIANTI Project. System Architecture Specification. Deliverable D3.1, August 2008.

[14] Consultative Committee for Space Data Systems. Space Communications Protocol Specification (SCPS) – Transport Protocol. CCSDS 714.0-B-1, available from http://www.scps.org/, May 1999.

[15] K. Fall. A Delay-Tolerant Network Architecture for Challenged Internets. In *Proceedings of ACM SIGCOMM 2003*, August 2003.

[16] T. Goff, J. Moronski, and D. Phatak. Freeze-TCP: A True End-to-end TCP Enhancement Mechanism for Mobile Environments. In *Proceedings of IEEE Infocom*, 2000.

[17] S. Guo, M. Falaki, E. Oliver, S. U. Rahman, A. Seth, M. Zaharia, U. Ismail, and S. Keshav. Design and Implementation of the KioskNet System. In *International Conference on Information Technologies and Development*, 2007.

[18] Z. J. Haas. Mobile-TCP: An Asymmetric Transport Protocol Design for Mobile Systems. 3rd International Workshop on Mobile Multimedia Communications, September 1995.

[19] T. Koponen, P. Eronen, and M. Särelä. Resilient Connections for SSH and TLS. *Proceedings of the Annual Technical Conference on USENIX'06 Annual Technical Conference*, 2006.

[20] M. Leech, M. Ganis, Y. Lee, R. Kuris, D. Koblas, and L. Jones. SOCKS Protocol Version 5. RFC 1928, March 1996.

[21] D. A. Maltz and P. Bhagwat. MSOCKS: An Architecture for Transport Layer Mobility. In *Proceedings of IEEE Infocom*, pages 1037–1045, 1998.

[22] Y. Mao, B. Knutsson, H. Lu, and J. Smith. DHARMA: Distributed Home Agent for Robust Mobile Access. In *Proceedings of the IEEE Infocom, Miami*, March 2005.

[23] R. Moskowitz, P. Nikander, P. Jokela, and T. R. Henderson. Host Identity Protocol. Internet Draft draft-ietf-hip-base-10.txt, Work in progress, October 2007.

[24] J. Ott. Application Protocol Design Considerations for a Mobile Internet. In *Proceedings of the 1st ACM MobiArch Workshop*, December 2006.

[25] J. Ott. Delay Tolerance and the Future Internet. In *11th International Symposium on Wireless Personal Multimedia Communications*, 2008.

[26] J. Ott and D. Kutscher. Drive-thru Internet: IEEE 802.11b for "Automobile" Users. In *Proceedings of IEEE Infocom, Hong Kong*, March 2004.

[27] J. Ott and D. Kutscher. The "Drive-thru" Architecture: WLAN-based Internet Access on the Road. In *Proceedings of the IEEE Semiannual Vehicular Technology Conference May 2004, Milan*, May 2004.

[28] J. Ott and D. Kutscher. A Disconnection-Tolerant Transport for Drive-thru Internet Environments. In *Proceedings of IEEE Infocom, Miami*, March 2005.

[29] J. Ott and D. Kutscher. Exploiting Regular Hot-Spots for Drive-thru Internet. In *Proceedings of KiVS 2005, Kaiserslautern, Germany*, March 2005.

[30] J. Ott, D. Kutscher, and M. Koch. Towards Automated Authentication for Mobile Users in WLAN Hot-Spots. In *Proceedings of VTC Fall 2005*, September 2005.

[31] V. N. Padmanabhan and J. C. Mogul. Using Predictive Prefetching to Improve World-Wide Web Latency. In *Proceedings of ACM SIGCOMM*, 1996.

[32] C. E. Perkins. (ed.) IP Mobility Support for IPv4. RFC 3344, 2002.

[33] J. Salz, A. C. Snoeren, and H. Balakrishnan. TESLA: A Transparent, Extensible Session Layer Architecture for End-to-end Network Services. In *4th Usenix Symposium on Internet Technologies and Systems*, March 2003.

[34] K. Scott and S. Burleigh. Bundle Protocol Specification. RFC 5050, November 2007.

[35] A. Seth, S. Bhattacharyya, and S. Keshav. Application Support for Opportunistic Communication on Multiple Wireless Networks. Manuscript available from http://blizzard.cs.uwaterloo.ca/keshav/home/Papers/data/05/ocmp.pdf, November 2005.

[36] P. Sinha, N. Venkitaraman, R. Sivakumar, and V. Bharghavan. WTCP: A Reliable Transport Protocol for Wireless Wide-Area Networks. In *Proceedings of ACM Mobicom, Seattle, WA, USA*, pages 231–241, 1999.

[37] A. C. Snoeren, H. Balakrishnan, and M. F. Kasshoek. Reconsidering Internet Mobility. In *8th Workshop on Hot Topics in Operating Systems, HotOS-VIII*, May 2001.

[38] F. Sultan, K. Srinivasan, D. Iyer, and L. Iftode. Migratory TCP: Highly Available Internet Services Using Connection Migration. Technical Report DCS-TR-264, Rutgers University, December 2001.