Secure Unified Cellular Ad Hoc Network Routing

Jason J. Haas
University of Illinois at Urbana-Champaign
Urbana, Illinois, U.S.A.
jjhaas2@illinois.edu

Yih-Chun Hu
University of Illinois at Urbana-Champaign
Urbana, Illinois, U.S.A.
yihchun@illinois.edu

Abstract*—Previous simulations have shown substantial performance gains can be achieved by using hybrid cellular and wireless LAN (WLAN) approaches [1]. In a hybrid system, a *proxy* in an area of strong connectivity (and therefore higher bandwidth) forwards traffic on behalf of a *client* in an area of weaker connectivity (and therefore lower bandwidth). The proxy routes traffic between the base station (over a cellular link) and the client (over a WLAN). Such approaches have had limited practical applicability due to substantial security risks, including eavesdropping, intentional performance degradation and cheating the incentive schemes. The Secure Unified Cellular Ad Hoc Network (SUCAN) protocol is designed to address these risks, allowing the deployment of hybrid networks. SUCAN uses incentives and cryptographic techniques to eliminate cheating by self-interested hosts, while limiting the damage caused by malicious hosts. We implemented SUCAN on Verizon's Broadband Access network. To our knowledge, this is the first real-world implementation of a hybrid cellular network protocol. Our implementation results show that the SUCAN system can provide substantial performance increases and protects against performance degradation even in the presence of malicious behavior.

I. Introduction

Modern cellular data services, such as CDMA2000 [2], adaptively choose modulation schemes and coding rates based on the signal-to-noise ratio between the base station and the mobile user, allowing for more efficient communications with mobile users that are closer to the base station, while still providing lower-speed coverage to users that are farther away from the base station. Future data services such as WiMAX [3] will make even more extensive use of dynamic bandwidth selection. *Hybrid networks* [1] can be used to increase the overall capacity of a network by relying on a *proxy* in an area of strong connectivity (and therefore higher bandwidth) to forward traffic on behalf of a *client* in an area of weaker connectivity (and therefore lower bandwidth). The proxy routes traffic between the base station (using its cellular link) and the client (using a wireless LAN (WLAN)).

The increased availability of devices with both WiFi and high data rate (HDR) cellular network connections [1] makes the use of hybrid network routing schemes more feasible, however, a prerequisite to successful deployment is the design of a *secure* hybrid network protocol that secures incentives given for participation and prevents an attacker from decreasing the performance of a victim. This paper describes our new protocol, the Secure Unified Cellular Ad hoc Network (SUCAN) routing protocol, that addresses these security concerns and secures against both selfish and malicious adversaries. Our protocol defends against noncompliance on two fronts. First, we use incentives and penalties to eliminate cheating by self-interested hosts, that is, by hosts that want to gain a performance advantage. Incentives are rewards given for correct participation in SUCAN. They can take different forms depending on what is beneficial for an individual node. Second, we use *grudging*, to limit the damage done by malicious nodes that do not care about negative impacts to their performance. Grudging is remembering previous unsuccessful encounters, so that if a node holds a grudge against another node, the former will no longer interact with the latter. Our implementation results show that the SUCAN system provides substantial performance increases in network efficiency and can provide substantial performance increases in throughput even in the presence of malicious behavior.

Wireless spectrum is in increasingly high demand for cellular carriers. Recently, the FCC held auction 73 for the 700-MHz spectrum band. The FCC received \$19.4 billion in provisionally winning bids, including \$9.4 billion from Verizon Wireless and \$6.6 billion from AT&T.[2] The amount of money spent in these auctions shows the value wireless spectrum has to cellular carriers. Thus, it is in a carrier's best interest to use their spectrum in the most efficient manner. Hybrid networks address this interest by moving data from areas of low bandwidth efficiency (low throughput) to areas of high bandwidth efficiency (high throughput) through another medium, resulting in more efficient use of cellular bandwidth. Hybrid networks can also be used to extend the range of cellular base stations and include regions of extremely poor connectivity (e.g., in urban canyons).

The rest of the paper is organized as follows. In Section II we discuss related work on which we build. We present SUCAN design details in Section III. We analyze the security of our protocol and implementation in Section IV. We present the results of our evaluation, implementation, and methodology in Section V. We conclude in Section VI.

II. Related Work

There are many other papers in the literature that study hybrid cellular and WLAN networks [4–6]. However, many

[1]e.g., Samsung SCH-i730, Apple iPhone, IBM T-60, and T-Mobile G1

[2]FCC auctions: Summary: Auction 73

TABLE I
EXPLANATION OF TERMS FOR FIGURES 1 AND 2.

Term	Meaning
C	The client node
P	The proxy or a possible proxy node
BS	The base station
*	The broadcast address
$E_K(X)$	Encryption of message X using key K
$MAC_\varepsilon(X)$	The MAC computed over message X using ε's key.
$TPut_{\varepsilon,\{up,down\}}$	ε's throughput in either the up or down direction.
Exp	The expiration time associated with the packet.

do not address security concerns and focus on performance issues, which we do not discuss because of space constraints.

Our protocol builds on the previous work of Luo et al. [1]. Simulations of their UCAN protocol show the effectiveness of hybrid networks for increased cellular capacity. UCAN proposes two ad hoc network routing protocols that can discover suitable proxies: a greedy protocol and an on-demand protocol. UCAN uses incentives for forwarding; each proxy is provided with additional performance based on the base station's knowledge of that proxy's participation in UCAN. However, UCAN does not address a few basic security concerns. First, an attacker can attack the Route Discovery process by lying about its throughput to the base station or not forwarding Route Discovery messages, thus preventing a client from discovering a good proxy. In addition, while downloading packets through a proxy, the proxy can alter or drop the packets, resulting in a man-in-the-middle attack or a denial-of-service attack. The authors simulated UCAN using NS-2 and showed that the throughput of deploying nodes can be significantly increased.

Another hybrid networking protocol that addresses security, JANUS [7], is also susceptible to attacks. JANUS relies on GPS data and references another position verification technique, Sat-Range[8], to assist in determining the location of nodes. The base station uses this information to estimate the link speed of the WiFi connections on each hop along the route from the client to the end proxy. Using this approach suffers from three problems. First, GPS signals may not always be available, especially inside buildings. Second, the position information given by the GPS is subject to modification by a malicious host when it reports its GPS data to the base station. Third, even if the position information is correct, there is no way for the base station to correctly infer the channel fading and noise from the position information of the end points of a link. For example, two links may report positions within a few meters of each other, which with line-of-sight would allow for maximum throughput on the WiFi channel. However, if line-of-sight be interrupted by walls or floors, the channel may not support sufficient throughput between the two nodes. The authors simulate JANUS in NS-2 to investigate the overhead in number of messages as compared to UCAN. While overhead is an important statistic, it addresses neither the short-term nor long-term utility of the system.

III. PROTOCOL DESIGN

In this section, we describe the SUCAN protocol in detail. We divide the description of the protocol into uplink and downlink behavior. The uplink protocol is used to upload data from the client to the cell network. Similarly, the downlink protocol is used to download data from the cell network to the client. Table I lists the terms we use in this section.

We use incentives, charges, and data integrity checks to prevent attacks such as man-in-the-middle, which could be performed by a malicious proxy by modifying or dropping packets destined for the client. Incentives are rewards credited to participants for good behavior. Incentive examples include increased queuing priority and free increases in a user's transfer quota. Misbehaving nodes can be charged to deter malicious and non-participatory behavior. When a victim node interacts with a malicious node, it remembers the identity of the malicious node, holding a grudge so that the victim will no longer interact with the malicious node. Thus, the grudged-against node can only do a limited amount of damage to the grudging node, even if the attacker is irrational. These three mechanisms are described in further detail in Section III-D. We avoid multi-hop routing problems by restricting the set of eligible proxies to direct neighbors, thus eliminating possible attacks against multihop routing.

A. System Model

a) Attacker Model: We assume each node shares a secret key with the base station, which is reasonable, since any node using the network must have a preexisting relationship with the cellular provider. Since SUCAN participants must be customers of the cellular provider, the base station can identify any participant, which the base station would need to do to charge for regular operation on the cell network. We do not consider physical layer attacks in this paper because attackers able to successfully mount physical layer attacks need not even fulfill the assumption that they are customers of the cellular provider. Other than the assumptions stated above, we place no restrictions on our attacker model. We discuss and analyze SUCAN performance when malicious nodes participate in Section III-D. We also give a more detailed discussion of the types of attacks an attacker could attempt on SUCAN in Section IV-B.

b) Interfaces: SUCAN requires two network interfaces at each participant. We assume a WLAN interface and a HDR interface exist at each node. It is possible to replace the WLAN interface with another type of network interface, such as Ethernet. However, with a WLAN interface, nodes are allowed to be mobile.

c) Internals: Since hybrid networks rely on the difference between achievable HDR data rates at the client and proxy, our protocol considers the *throughput* at both the client and every available proxy.

In the current design of SUCAN, only a neighbor can serve as the proxy, thus limiting potential denial-of-service attacks on the ad hoc network. SUCAN is a first step toward securing hybrid networks, and extensions could also allow for multihop routing.

SUCAN exchanges keys between participants as described below. The exact mechanism for key exchange is orthogonal to this paper, thus we exclude key exchange specifics.

$C \rightarrow *$: $\langle$ Route Request, C, $\text{TPut}_{\text{C,up}} \rangle$
$P \rightarrow C$: $\langle$ Route Reply, P, $\text{TPut}_{\text{P,up}} \rangle$
$C \leftrightarrow P$: Key Exchange
C: $\text{Auth}_{i,\text{BS}} = \text{MAC}_{\text{BS}}(\text{ETT}(\text{Pkt}_i), \text{P}, \text{Pkt}_i)$
$C \rightarrow P$: $\langle \text{UpData}_i, \text{ETT}(\text{Pkt}_i), \text{P}, \text{Pkt}_i, \text{Auth}_{i,\text{BS}} \rangle$
$P \rightarrow BS$: $\langle \text{UpData}_i, \text{ETT}(\text{Pkt}_i), \text{P}, \text{Pkt}_i, \text{Auth}_{i,\text{BS}} \rangle$
$BS \rightarrow C$: $\langle \text{Ack}(\text{Pkt}_i) \rangle$

Fig. 1. Uplink protocol packet exchange example.

B. Uplink Protocol

A node initiates the uplink functionality of SUCAN when it has data to upload. Figure 1 shows the messages sent as part of the uplink protocol. First, the client broadcasts a Route Discovery packet on its WLAN interface. A Route Discovery packet contains the client's identity and its upload throughput. When a neighbor receives the Route Discovery packet, it compares the client's throughput with its own. If the potential throughput improvement is sufficiently large, the neighbor composes and replies with a Route Reply packet, which contains its identity and its upload throughput. Based on the replies received by the client, the client (potentially) chooses a proxy and establishes a shared secret key with the proxy.

For each packet in the client's HDR queue, the client estimates the transmission time of that packet based on the rate currently experienced by the client. Whenever the client sends a packet to a proxy for forwarding, it includes with that packet a SUCAN header that includes the expected transmission time (ETT) of that packet, the identity of the proxy, and an authenticator covering the data packet and the SUCAN header for the base station. We use Message Authentication Codes (MACs) as our authenticators in SUCAN.[3]

The client does not assume that forwarded packets are successfully delivered to the base station but leaves these packets in its queue, marked so that they will not be sent before their ETTs. If the client receives an acknowledgment for such a packet from the base station prior to the ETT, the client removes the packet from its HDR queue.

We include the ETT so the proxy can determine whether or not the packet can be successfully proxied. When the ETT expires, the proxy assumes that the client will send the packet itself. As a result, if the proxy thinks it cannot send the packet in time, it drops the packet. If the proxy believes the packet can be successfully proxied, based on its current bandwidth, queue length, and the expected latency from the base station to the client, then the proxy forwards the packet to the base station. A proxy is credited only for the forwarding when the base station delivers an acknowledgment to the client prior to the expiration of the ETT.

Upon receiving a proxied packet, the base station checks the authenticator to ensure that the packet was sent from the client, that the ETT has not yet expired, that the proxy was actually chosen by the client (based on the proxy address specified in the packet), and that the packet has not been modified. If the checks pass, the base station decapsulates the packet, treats it as if the packet had come directly from the client, and acknowledges the transmission directly to the client. The acknowledgment is sent over the HDR to reduce the proxy's uncertainty of acknowledgment latency.

[3]When we use the acronym MAC, we mean message authentication code and not medium access control.

$C \rightarrow *$: $\langle$ Route Request, C, $\text{TPut}_{\text{C,down}} \rangle$
$P \rightarrow C$: $\langle$ Route Reply, P, $\text{TPut}_{\text{P,down}} \rangle$
$C \leftrightarrow P$: Key Exchange
C: $\text{Auth}_{i,\text{BS}} = \text{MAC}_{\text{BS}}(\text{TPut}_{\text{P}}, \text{C}, \text{P}, \text{Exp})$
$C \rightarrow P$: $\langle \text{Authorize}, \text{TPut}_{\text{P}}, \text{C}, \text{P}, \text{Exp}, \text{Auth}_{i,\text{BS}} \rangle$
$P \rightarrow BS$: $\langle \text{Authorize}, \text{TPut}_{\text{P}}, \text{C}, \text{P}, \text{Exp}, \text{Auth}_{i,\text{BS}} \rangle$
$BS \rightarrow P$: $\langle \text{Download Data}_m, \text{E}_K(\text{Pkt}_m) \rangle$
$P \rightarrow C$: $\langle \text{Download Data}_m, \text{E}_K(\text{Pkt}_m) \rangle$
$C \rightarrow P$: $\langle \text{Ack}(\text{Download Data}_m) \rangle$
$P \rightarrow BS$: $\langle \text{Ack}(\text{Download Data}_m) \rangle$
$BS \rightarrow C$: $\langle \text{Key}, K \rangle$

Fig. 2. Downlink protocol packet exchange example.

C. Downlink Protocol

Figure 2 illustrates the message exchange described below. As in the uplink protocol, a client node initiates the downlink protocol by broadcasting a Route Discovery packet on its WLAN interface, which proceeds, including key exchange, as described in the previous section. As in the uplink protocol, the client chooses a proxy based on the potential improvement in throughput, and performs a key exchange protocol with the chosen proxy. The client then sends an authorization packet to the proxy node, containing the proxy's advertised throughput, the client and proxy's identities, the expiration time after which the authorization is no longer valid, and an authenticator for the base station. Thus, the proxy can verify it has been chosen by the client. When the proxy forwards this packet to the base station, the base station can verify the proxy was chosen by the client.

When the base station receives the authorization, it verifies the information included in the authorization, including proxy download throughput and identity. The base station checks the authenticator to ensure the authorization was generated by the client. After verifying this information, the base station chooses a random session key K. Each packet sent to the client through the proxy is encrypted with this session key. All information that uniquely identifies each packet (e.g., packet number, TCP sequence number) is encrypted in this manner. Thus, neither the client nor the proxy can generate false acknowledgements. The proxy will not be able to generate acknowledgements because the proxy cannot correctly identify the original packet, so the proxy cannot gain unearned incentives through false acknowledgements. The client, similarly, cannot generate a false acknowledgement before both receiving the packet and receiving the key from the base station. For each packet in its queue, the base station maintains an ETT, and selects packets for relaying through the proxy based on packets' ETTs and the expected round trip time through the proxy to the client. Packets sent through the proxy are encrypted to provide confidentiality to the client and force the client to acknowledge the packet, allowing the proxy to be credited. Since download packets are encrypted and the proxy does not have the key, the proxy can not generate false acknowledgements and send them to the base station. The proxy forwards the encrypted packets on its WLAN interface to the client.

When the client receives a packet from the proxy, it returns an acknowledgment. If the proxy receives the acknowledgment before the ETT of the corresponding packet, it relays the acknowledgment to the base station. If the base station has not previously received an acknowledgment for this session, it sends the key directly to the client. By sending the key directly to the client, the client receives the same level of privacy it would have had if it had not used the proxy. We do not consider the proxy sniffing this key as it is transmitted over the HDR, as that would provide no worse confidentiality than if the client used the HDR alone, and sniffing another user's traffic in this manner is illegal in many places. Furthermore, because the key is only sent once per session, the overhead is limited. By sending the key only after the first acknowledgement is received by the base station, the client is forced to participate correctly at least before the key is sent to it. Thus, the network can enjoy sufficient benefit if the client participates, or the client receives no benefit if it does not participate. The key K is used for the duration of the time that the client uses the proxy and protects against a client not acknowledging packets before K is released to the client.

D. Incentives and Charges

SUCAN uses incentives to encourage network nodes to participate and uses charging to discourage self-interested malicious behavior. Charging can be either monetary, which is most suitable when customers pay by amount of traffic sent, or can affect queuing priorities, which is most suitable when customers pay a flat rate for an unlimited amount of service.

Many SUCAN headers require an authenticator for the base station. If the base station reports to a proxy that the client's authenticator was incorrect, the proxy knows the packet was sent by the client and the client deliberately included an incorrect base station authenticator. The proxy will then bear a grudge against the client and never interact with that client again.

Client nodes using SUCAN benefit from higher upload and download performance. When data is uploaded through a proxy, the proxy has the most knowledge of whether or not the data is likely to be sent to the base station in time. In this case, the proxy assumes all the risk for upload packets (in the event that the upload is redundantly uploaded because the acknowledgment is not delivered to the client prior to the expiration of the ETT). Because the proxy bears all the risk, it should be credited with a substantial portion of the saved uplink bandwidth. Except for the aforementioned fake-authenticator attack, neither clients nor proxies need remember when they are harmed as part of the upload process because the client only receives a net benefit, and the proxy makes decisions based on more information than what the client has. Thus, the proxy has more responsibility to deliver the packets than the client, since the ETT is sent in each packet and the client authenticates the ETT.

When a client and proxy cooperate for more efficient download performance, the client and proxy evenly split any rewards received from the saved bandwidth. However, because the client and proxy can harm each other, as described in the following paragraph, each can hold a grudge against any node that has previously harmed it. Grudging ensures that the amount of damage an attacker can inflict on a victim is bounded, even if the attacker is willing to pay the same amount as the victim. The identity information exchanged as part of Route Discovery is used to exclude unqualified clients or proxies.

There are two failure modes when downloading. First, the client can stop sending acknowledgments for successfully forwarded packets. Second, the proxy can stop forwarding packets sent by the base station. Because these two events are indistinguishable to the base station, both the client and the proxy are charged for failures. To reduce the time during which the client and proxy are at risk, the base station maintains a *window* of packets sent to the proxy that have not yet been acknowledged. This window is set to some *initial window size*. The proxy is used because it offers better performance than sending directly to the client; we denote this increase in performance as a factor of α (the client would have $\alpha = 1$ when using itself as a proxy). Our protocol headers result in some overhead, so the size of each packet delivered through SUCAN is larger by a factor of $\beta > 1$ ($\beta = 1$ would mean no overhead). Each successfully delivered packet saves an amount of bandwidth given by $\gamma = \frac{\alpha}{\beta}$. For example, a client may experience a 100 kbps throughput and a proxy 1000 kbps throughput. Thus, $\alpha = \frac{1000 \text{ kbps}}{100 \text{ kbps}} = 10$. Suppose the amortized overhead of using SUCAN is 10%. Thus, $\beta = 110\% = 1.1$. Therefore, when the client uses this proxy, the bandwidth efficiency increases by a factor of $\gamma = \frac{\alpha}{\beta} = \frac{10}{1.1} = 9.1$, that is, the bandwidth of the cell network is used 9.1 times more efficiently for packets routed through the proxy compared to what it would have been without SUCAN. After the first window is completely acknowledged, the new window size is set to γ times the initial window size, so that if the entire second window were lost, there would be no reduction in overall network performance. Furthermore, whenever a complete window is acknowledged, the window size can be increased by a factor not more than γ, allowing for exponential growth of the window size. More conservative growth may be necessary to ensure that a specific client does not experience reduced network performance, for example as a result of TCP's reaction to loss; these impacts are described in Section V-B. Larger window sizes improve performance only when the additional space is used; thus, a window's size should not be increased unless there are packets that can be sent as a result.

When the client uses a proxy for upload, and the proxy fails to forward a packet from the client, the proxy receives no benefit, and the client transmits the packet on the HDR, incurring no reduction in performance. The proxy incurs some risk, such as the risk of packet loss, which is offset by the likely benefit, should the upload succeed. To provide maximum benefit to the network, after each successful proxy upload, the base station should schedule the proxy more often and the client less often, so that after the proxy successfully uploads enough packets, the base station relies almost entirely on the proxy for upload. An uploading client should maintain a window similar to that used by the base station for downloading, as described above, to ensure that its net bandwidth is increased should the proxy fail or maliciously discard packets.

A group of high-bandwidth attackers can attempt to defraud the carrier by pretending to be a set of low-bandwidth clients and high-bandwidth proxies. In this case, the base station rewards proxies with even more time slots, thereby causing unfairness to other network users. To defend against this attack, we suggest that the carrier should perform *time-fair* scheduling at the base station. For example, if an attacker controls two users, one at $a = 1$ Mbps and one pretending to have a reduced rate of $b = 100$ kbps, then the base station provides equal time slices to each of the two users (e.g., one slot per frame). When the proxy forwards a packet, the slots that would otherwise be allocated to that packet are divided. $\frac{b}{a}$ of those slots (roughly one-tenth in this example) are required to transmit the packet; the remaining slots can be divided between rewards for the proxy and the client, and some excess network capacity saved for the network. In other words, the client is charged a number of time slots equal to the number of time slots that it *would have taken* if the packet were directly sent over the HDR, less some fraction that is returned as a reward for the client. In this case, a colluding client and proxy together receive fewer time slots than the same client and proxy without collusion (except that the client's slots are used more efficiently, improving the performance of a client that is not cheating).

E. Long-Term Behavior

SUCAN is most vulnerable to attack in the early stages, before the first window has been successfully delivered because after the first window has been successfully delivered, future losses do not adversely impact network performance. If a malicious proxy initiates an attack later in the data transfer stage, the network and client have already achieved a sufficient benefit from the proxy's good behavior up until that point so that any misbehavior afterward does not result in any net harm. For example, on a long file transfer to the client, the proxy might choose to start dropping packets after it has sent a couple of megabytes of data. By this time, the client has received significantly higher throughput than what it was able to receive itself, and the network has similarly improved aggregate system-wide bandwidth by using the proxy instead of the client.

As the benefit of using the proxy continues to accrue, at some point even the loss of, e.g., one or two entire TCP windows will still result in a net benefit. At such times, the base station may choose to relay all traffic through the proxy, and only stop doing so when one or two entire windows are lost, which benefits TCP because it behaves best when sent over a single channel [9], and benefits the network provider because every packet exchanged with directly the client results in reduced network throughput.

F. Implicit Acknowledgments

When TCP is used, TCP acknowledgments can be used as implicit SUCAN acknowledgments, particularly when downloading. Specifically, when all TCP download traffic traverses the proxy, then each TCP acknowledgment can serve as an acknowledgment for one or more packets sent over the proxy. If the TCP acknowledgments are sent directly over the HDR, the proxy knows the client continues to send acknowledgments because packets continue to flow to the proxy, meaning the base station is still willing to use the proxy.

IV. Security Analysis

A. Prevention Mechanisms

The MAC (keyed with the session key K) covering uploading packets provides data integrity and allows the base station to detect tampering with proxied upload packets. The base station attaches a similar MAC covering packet data on the download side to provide similar assurances and tamper detection.

SUCAN acknowledgments mitigate attacks where a proxy node drops packets sent by or to the client after the route has been established. In this attack, drops reduce the number of acknowledgments returned to the sender, that is, returned to the base station when downloading and to the client when uploading. When an acknowledgments is not returned, the sender assumes the packet has been lost, retransmits the corresponding packet over the HDR, and reduces the window size for the proxy accordingly.

We include a 64-bit timestamp in the upload data packet header to allow the proxy and the base station to decide whether to forward the data packet or to drop it, and to prevent replay attacks. We need to defend against replay attacks because a successful replay attack can cause a proxy to gain unearned benefits from the network provider.

B. Attacker Behavior

We assume an attacker of SUCAN can be Byzantine in behavior. An attacker, therefore, may choose to attack any piece or stage of SUCAN. There are four main types of attacks against SUCAN. First, the attacker could try to exhaust client or base station resources such as energy and CPU time. Second, the attacker could try to reduce the performance of a legitimate node, e.g., by serving as the proxy for that node but not forwarding packets. Third, the attacker could try to unfairly increase its own performance, e.g., by using a victim as a proxy but not providing the victim with any credit for forwarding. Finally, the attacker could try to cheat the incentive scheme, either by gaining incentives which he does not deserve or depriving a proxy of its rightful incentives.

A number of preexisting key exchange protocols can be used in conjunction with SUCAN. Because of the abundance of existing, well-established, and applicable key exchange protocols, we do not attempt to build a new key exchange protocol. We rely on the existing published protocols for key exchange. In particular, because we use the cellular provider as a central point of trust, the provider can also serve as a Certificate Authority. Certificates can be exchanged during the route establishment phase of SUCAN by appending them to Route Request and Route Reply packets. Computational Denial-of-Service can be mitigated through the use of cryptographic challenges disseminated by the base station, using a scheme such as Portcullis [10]. Alternatively, valid public keys can be placed into a Merkle Tree [11].

In the first attack, the attacker generates SUCAN traffic in order to consume energy and CPU time of a victim machine. When the attacker initiates SUCAN, a victim that has insufficient CPU, battery, or memory resources can simply ignore the attacker, since SUCAN is an optional protocol that can in many cases provide mutual benefit to the sender and the proxy. However, a proxy having insufficient resources cannot benefit by acting as the proxy, so any node ignoring the SUCAN protocol does not affect its security properties. Alternatively, the attacker can attempt to exhaust the CPU or battery resources of a victim that has itself initiated the use of SUCAN; that is, when the victim is searching for an SUCAN proxy. However, the computational overhead of SUCAN is quite light, which we have also determined through analysis and measurement. Due to space restrictions, we omit presenting the results of our analysis of SUCAN's resource usage.

In the second attack, the attacker tries to subvert SUCAN by reducing the performance of a legitimate client. It can do this either by attacking Route Discovery or the forwarding portion of the protocol.

During Route Discovery, an attacker may choose to drop Route Requests, but this is no different than a node not participating in SUCAN. A more subtle attack at this stage of the protocol would be for an attacker posing as a proxy to falsely claim its throughput. If the attacker under-claims its throughput, this is no more severe than the attacker that drops Route Requests because the attacker will be chosen as a proxy less often and only when the proxy's throughput is much better than the client's throughput. If an attacker over-claims its throughput and is chosen by the client, the base station will detect this attack and inform the client that the proxy is not suitable, since the base station can verify the proxy's claimed throughput.

After Route Discovery is completed and packets are being proxied (uplink or downlink), a malicious proxy could alter or drop packets sent through it. If the proxy behaves in such a manner, timeouts will occur at either the client or the base station because packets are not discarded after they are given to the proxy; rather, they are held-back until an acknowledgment is received. If the packet is modified, the packet's MAC will be invalid, so the packet will be dropped. Therefore, whether the packet is dropped or modified, no SUCAN acknowledgment will be sent; after the retransmission timer expires, the packet will be released directly to the other party. Furthermore, because the timeouts of these packets are chosen such that they will expire before the packet would have otherwise been transmitted, the service experienced by the client is no worse than what it would achieve without SUCAN. With this attack, a proxy receives no incentives from dropped packets since an acknowledgment is required for incentives to be assigned. A malicious proxy could introduce extra delays in forwarding packets. If the delay causes the packets to be received before the timer expires, then the attack improves performance, otherwise the attack is exactly the same as if the attacker simply dropped the packets.

In the third attack, the attacker participates as a SUCAN client and can, for example, under-claim its throughput to use a proxy having lower throughput. If the attacker continues to receive from the base station at high data rates, the base station can detect this attack by comparing the under-claimed rate with the rate the client has; otherwise, this is indistinguishable from an attacker that simply moves farther away from the base station.

In the fourth attack, the attacker either acts as a client or a proxy and tries to subvert the incentive system. When the attacker is established as a SUCAN client, the attacker may refuse to return acknowledgments to the base station, to deny the proxy its rewards. However, because the acknowledgments are sent through the proxy instead of directly to the base station, the proxy can detect lacking acknowledgments. In this case, the proxy grudges against the client. When the base station does not receive acknowledgments for packets sent to a malicious proxy for forwarding, the proxy will not receive credit for forwarding those packets. In general, the base station will neither give incentives for bad behavior, nor will it distribute incentives incorrectly because each packet is sent through a particular proxy, and each proxy is rewarded for packets that are sent through it.

SUCAN is built under the assumption that an attacker cannot inject packets over the HDR unless those packets bear a correct source and destination. This assumption is reasonable in many instances because most cellular hardware ensures that correct headers are sent for each packet sent over the HDR. When a more powerful attacker is contemplated, our protocol can be modified to enforce this restriction using cryptographic means. In particular, each transmission over the HDR is encrypted using the key shared between the base station and the mobile, authenticated using a MAC computed using the same key, and includes a timestamp for replay prevention. These mechanisms ensure security even under a more powerful attacker.

In order to justify our statements regarding the first and second attacks, we developed an experimental methodology to evaluate the effectiveness of our defense mechanisms.

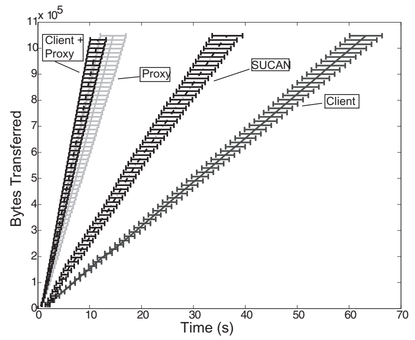


Fig. 3. TCP throughputs.

V. RESULTS

We used a single client and proxy in our experiments. Both the client and proxy used Kyocera KPC650 PCMCIA 1xEV-DO cards from Verizon for the HDR and Netgear WG511U 802.11a/b/g WiFi PCMCIA cards for the WLAN. A remote server ran the base station part of our software and web server software. We implemented SUCAN as three separate elements in the Click Modular Router [12] version 1.5.0: client, proxy, and base station. We adapted C/C++ AES code.[4] Our HMAC-SHA-1 functions were based on Click's SHA-1 support, which we used for creating MACs. We omit further details of our implementation due to space constraints.

Though we implemented and tested our upload protocol, limitations in the design and implementation of existing 1xEV-DO networks did not permit our protocol to show any improvements in upload speed. In particular, the 1xEV-DO specification has only five different rates for the "reverse channel" from the mobile to the base station, and all of these rates use the conservative BPSK modulation scheme [13]. At our test locations, we were unable to achieve data rates other than the highest rate under reliable coverage conditions. Consequently, we do not present any upload results. Future cellular data standards such as WiMAX [3] and 1xEV-DV use significantly more aggressive reverse channel modulation, and should be able to benefit substantially from our contributions.

We measured throughput with and without SUCAN using both TCP and UDP. We measured TCP performance by downloading a 1 MB file over HTTP from the base station computer. We measured UDP performance by having the base station flood the client with UDP packets for 10 s; netcat was invoked every 100 ms in order to keep the link saturated for the duration of the measurement. We omit further discussion of UDP because of space restrictions and because the UDP results simply show the channels being flooded, which results in the client always achieving throughput greater when it uses a proxy than what it could achieve individually, independent of how many packets the proxy drops.

A. Metrics and Presentation

We now describe how we present our measurements. Each graph shows time elapsed on the x-axis and bytes transferred on the y-axis. Each scenario is represented by a single series. For each run, we then determined the times at which a certain number of bytes had been successfully transferred, and computed the mean and the 95% confidence interval for these times across all of our runs for each scenario. Because our TCP measurements run for a fixed number of bytes, the metric measured at each intermediate milestone (intermediate number of bytes transferred) is *elapsed time*, thus our confidence intervals cover an amount of time (x-axis) and the error bars are horizontal.

We choose to evaluate our system based on throughput provided to the client. We do not present measurements of saved bandwidth for network providers because as mentioned above, after the initial window of packets is exchanged, every packet successfully sent through the proxy provides only a net gain for the network provider, and the network provider should only schedule packets to be sent through the proxy, so that even if the proxy would drop the packets and the packets would need to be resent directly to the client, there would be a net gain in terms of bandwidth saved. Throughput allows us to view the short-term and long-term performance of the reputation based system from a client's perspective. In general, if a proxy can behave correctly initially but can change and drop packets long-term, then the reputation system is susceptible to attack. SUCAN is designed to mitigate the behavior of malicious proxies that act in this manner.

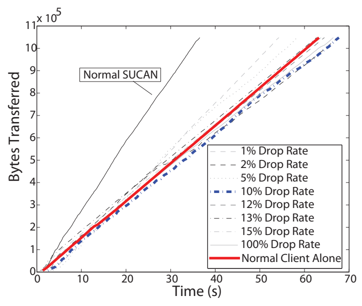
B. TCP Performance

Figure 3 shows the number of TCP bytes transferred plotted against time for flows to the client directly, the client with SUCAN, the proxy directly, and a theoretical plot of client plus proxy throughput. Higher lines and larger slopes indicate better network performance. The graph shows that SUCAN provides a net increase to the client's throughput. However, SUCAN performs worse than the proxy's normal data rate due to four factors: (1) the overhead of encapsulating packets, (2) the limitations on our ability to change the scheduler at the cell provider base station, (3) the use of multiple channels reduces the throughput of TCP as discussed in Section III-E, and (4) the windowing behavior specific to our implementation causes lower throughput earlier during a session than what would be theoretically possible with SUCAN.

C. Malicious Behavior Performance

Figure 4 shows TCP performance for a malicious proxy dropping 1%, 2%, 5%, 10%, 12%, 13%, 15% and 100% of packets. The two plots are the full transfer and a zoomed-in version. The 10% drop rate was the most effective for two reasons. First, at low drop rates, the client obtains a significant performance improvement from SUCAN, and continues to use the proxy, resulting in the client obtaining better throughput than its own. Second, at higher drop rates, the proxy's window eventually shrinks to zero, so no more packets are proxied. We use a 10% drop rate for the remainder of our analysis of malicious behavior. The 10% drop rate is the most effective rate for a malicious proxy to drop packets because of our implementation and is not inherent in the design of

[4]see http://fp.gladman.plus.com/AES/index.htm



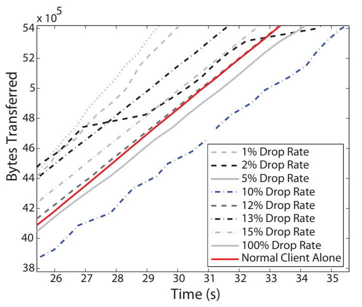


Fig. 4. Performance under various malicious drop rates.

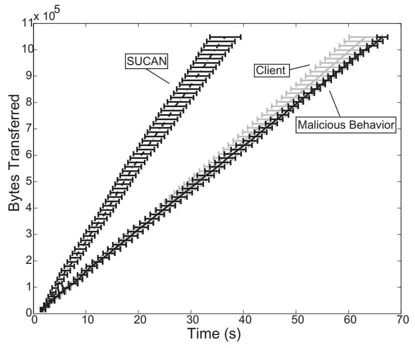


Fig. 5. TCP throughputs with and without a malicious proxy.

SUCAN. Asymptotically, with our implementation, a proxy must maintain a drop rate less than 12.5%, as at 12.5%, 1 in 8 packets will be dropped statistically, and the proxy's window will evaporate due to our choice of initial/minimum window size being 8. 10% turns out to be the best choice for malicious behavior, if the proxy is randomly dropping packets at a given rate. If the proxy chooses a lower rate, the client receives a significant benefit from the packets that are successfully proxied. If the proxy chooses a higher rate, either statistical variation causes the proxy to be dropped, or the proxy is asymptotically dropped, if the rate is higher than $\frac{1}{\text{initial window size}}$.

We did not analyze different proxy drop rates for UDP because UDP performance does not degrade due to loss. Since UDP floods the channel, any packets passed by the proxy only increase the client's perceived performance, and the transport layer neither tracks losses nor adjusts the offered load in response to such losses.

Figure 5 shows the number of TCP bytes transferred plotted against time for flows to the client directly, the client with SUCAN, and the client using a malicious proxy. The malicious behavior clearly removes any performance gains that would be experienced by using SUCAN, but the performance is not significantly worse than if the client had not used SUCAN at all. The slight performance degradation is caused by the overhead of our implementation of SUCAN.

VI. Conclusions

We have presented SUCAN, a new hybrid networking protocol. SUCAN uses incentives, charging, grudging and security primitives to protect participating nodes from malicious nodes, encourage participation in SUCAN, and provide a limited level of data integrity and confidentiality. Cellular network providers would benefit from SUCAN due to the bandwidth saved from using a more strongly connected proxy rather than directly using a more weakly connected client.

We have shown through analysis and experimentation that this protocol can provide substantial improvements in cellular network throughput, even when run in suboptimal conditions. Our protocol is designed to allow for no reduction in performance whatsoever, even in the case of malicious proxies. In our experiments, which were run without any control of the cell tower scheduler, we have shown that even in the case where a client selects a malicious proxy, its throughput to the network is only slightly worse than what it would have achieved on its own. Our results, to our knowledge, show the first real-world implementation of a hybrid networking protocol, and show the viability of hybrid networking protocols.

References

[1] H. Luo, R. Ramjee, P. Sinha, L. Li, and S. Lu, "UCAN: a unified cellular and ad-hoc network architecture," in *Proceedings of the 9th Mobicom*, 2003.

[2] D. N. Knisely, S. Kumar, S. Laha, and S. Nanda, "Evolution of wireless data services: IS-95 to cdma2000," *IEEE Communications Magazine*, vol. 36, pp. 140–149, Oct. 1998.

[3] C. Eklund, R. Marks, K. Stanwood, and S. Wang, "IEEE standard 802.16: a technical overview of the WirelessMAN$^{\text{TM}}$ air interface for broadband wireless access," *Communications Magazine, IEEE*, vol. 40, pp. 98–107, 2002.

[4] I. Ioannidis, B. Carbunar, and C. Nita-Rotaru, "High throughput routing in hybrid cellular and ad-hoc networks," *wowmom*, vol. 1, pp. 171–176, 2005.

[5] M. J. Miller, W. D. List, and N. H. Vaidya, "A hybrid network implementation to extend infrastructure reach," tech. rep., University of Illinois — Urbana–Champaign, 2003.

[6] N. B. Salem, L. Buttyán, J.-P. Hubaux, and M. Jakobsson, "A charging and rewarding scheme for packet forwarding in multi-hop cellular networks," in *MobiHoc '03: Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, (New York, NY, USA), pp. 13–24, ACM, 2003.

[7] B. Carbunar, I. Ioannidis, and C. Nita-Rotaru, "JANUS: towards robust and malicious resilient routing in hybrid wireless networks," in *Proceedings of 2004 ACM workshop on Wireless Security*, 2004.

[8] E. Gabber and A. Wool, "On location-restricted services," *IEEE Networks Magazine*, 1999.

[9] H. Han, S. Shakkottai, C. Hollot, R. Srikant, and D. Towsley, "Overlay TCP for multi-path routing and congestion control."

[10] B. Parno, D. Wendlandt, E. Shi, A. Perrig, B. Maggs, and Y.-C. Hu, "Portcullis: protecting connection setup from denial-of-capability attacks," *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 4, pp. 289–300, 2007.

[11] R. C. Merkle, "Protocols for public key cryptosystems," *sp*, vol. 00, p. 122, 1980.

[12] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek, "The click modular router," *ACM Transactions on Computer Systems*, vol. 18, no. 3, pp. 263–297, 2000.

[13] K. Etemad, *CDMA2000 EVOLUTION System Concepts and Design Principles*. John Wiley & Sons, 2004.