Photonic Network Communications, 4:1, 19–35, 2002

Integrating Security in the MAC Layer of WDM Optical Networks*

Borislav H. Simov
Enterprise Systems Technology Lab, Hewlett-Packard Co., Cupertino, CA 95014, USA
E-mail: boris_simov@hp.com

Jason P. Jue
Center for Advanced Telecommunications Systems and Services, University of Texas at Dallas, Richardson, TX 75083-0688, USA
E-mail: jjue@utdallas.edu

Srini Tridandapani
Department of Radiology, University of Michigan, Ann Arbor, MI 48109, USA
E-mail: srinit@umich.edu

Received June 13, 2001; Revised August 24, 2001

Abstract. We introduce a new technique for providing security in a broadcast-and-select, wavelength-division-multiplexed (WDM) optical network. The approach provides privacy of communications by employing a novel challenge-response scheme and exploiting the tuning delay inherent in optical receivers. The proposed technique can be integrated easily into any existing WDM media-access-control (MAC) protocol that employs tunable receivers. The modified protocol would require every idle user, who is not scheduled to receive data, to tune in to a channel that does not contain sensitive data. A violation of the protocol can be detected with very high probability, and appropriate measures can be taken against the violator. The technique provides features that cannot be achieved with cryptography alone. Significant benefits of the proposed approach include the ability to detect security violations as they occur, and an efficient mechanism to provide privacy for multicast transmissions.

We develop two simple solutions to deal with different levels of attack: (1) eavesdroppers working alone, and (2) eavesdroppers working in collaboration. We also introduce a dynamic channel allocation scheme that can further reduce the number of required overhead channels, with minimal loss in the capability to detect eavesdropping violations.

Keywords: challenge-response, cryptography, multichannel systems, optical channels, privacy, randomized protocols, security, tunability delay, wavelength division multiplexing

1 Introduction

The Internet is rapidly becoming a major world marketplace, with millions of financial transactions taking place daily. As the volume of financial data and other sensitive information being transmitted over the Internet continues to grow, the need for security is greater than ever. A network must provide adequate safeguards in order to ensure that sensitive data is not available to parties other than the destination. Security is especially a concern in local area networks that utilize a broadcast medium, such as a WDM optical network based on a passive-star coupler.

For the most part, existing security solutions rely primarily on cryptography [1, 2]. With encryption, it is assumed that anyone could intercept the encrypted message, but decrypting the ciphertext will be impossible without the secret decryption key. A completely different paradigm for providing security involves designing the physical and media access layers of the network in such a way that untrusted users are prevented from intercepting the message in the first place. In this paper, we will investigate such an approach to providing privacy at the media access layer. Our method relies on the fact that the physical hardware in a WDM local area network gives us some

* An earlier version of this paper was presented at IEEE ICC '98, Atlanta, GA, June 1998.

control over the access to the channels. To the best of our knowledge, very little work has been done to incorporate privacy at the physical or media-access layer [3, 4]—our paper is the first to integrate privacy and intrusion detection at the media-access layer.

In this paper, we consider a broadcast-and-select wavelength division multiplexed (WDM) optical network and a corresponding media access protocol that allows N users to share K data channels, where $K \leq N$. Each user has a transmitter, which may either be permanently fixed to a single channel or dynamically tunable over a range of channels, and a tunable receiver, which can be tuned to any channel. We assume that the optical fiber is secure against outside intrusion. Moreover, we do not expect an attack that uses additional devices to tap into the data channels. The question is whether, under these assumptions, privacy can be guaranteed. Unfortunately, the following problem may occur. Let Alice, Bob and Eve be three of the N users in the system and suppose Alice is scheduled to send a message to Bob. At the same time Eve is idle, that is, she is not scheduled to receive any data. Then Eve may decide to tune her receiver to the channel on which Alice is sending a message to Bob. Thus Eve can perform an unauthorized read, and therefore the system does not guarantee privacy.

How can we deal with the problem described above? As already mentioned, pre-existing solutions involve cryptography [1, 2, 5]. Alice and Bob agree on a secret key in advance and then use it to encrypt and decrypt the messages before and after the transmission, respectively. However, there are a few drawbacks to cryptographic schemes. First, there is a time overhead for encrypting and decrypting messages between Alice and Bob. This processing delay may not be tolerable for certain types of real-time traffic. Second, key management may be a non-trivial task. For example, key distribution may require either additional secure channels or long computations, depending on whether private or public key cryptography is used, respectively. In addition, keys may be forgotten, compromised, or may need to be renewed over time, which adds to the complexity of the problem. Furthermore, when providing security for multicast communications, the problem of distributing and revoking keys becomes an extremely complex issue, especially if the multicast group membership is changing dynamically. Finally, there is no easy way to detect an attack on the privacy of a message. If the key is unknowingly compromised, Alice and Bob will not be able to detect that their encrypted messages are being compromised.

In this paper, we suggest a novel approach to privacy which does not use encryption; however, it may be combined with encryption to provide an even higher degree of privacy. In addition to the existing K data channels, we introduce a challenge channel and a response channel on which no private data is transmitted. The main idea is that every time Eve is idle (i.e., not scheduled to receive any data) she is required to read on the challenge channel, thus she is prevented from eavesdropping on a data channel. At time slot t, a random bit string is sent to Eve on the challenge channel. Since Eve has only one receiver, she has to tune to the challenge channel; thus, she is not able to tune to a data channel and eavesdrop. On the next time slot, $t + 1$, Eve must send back the same random bit string that was sent to her as a challenge in time t, otherwise she is in violation of the protocol.

In most existing security mechanisms, security violations are discovered after the fact, when a significant amount of damage has already been done. In the proposed scheme, violations are detected with high probability as they occur, allowing for immediate action to protect the system and to discipline the perpetrator. Once a violation is detected, a number of actions may be taken. An extreme approach is to switch off all communications in the network, thus avoiding further compromise of sensitive information. Another option is to halt only the sensitive transmissions, and to resume transmission once compliance with the protocol is detected over some arbitrary number of time slots. A third option is to initiate higher-cost, encrypted transmission until compliance with the protocol is verified. Finally, off-line administrative actions against the perpetrators may be also considered.

One significant advantage of the proposed scheme is that it can provide privacy for multicast communication in an efficient manner. The trusted users in a multicast group can tune their receivers to the multicast channel, while untrusted users are simply forced to tune their receivers to a different channel. No significant modifications to the protocol are necessary to support multicasting.

Note that the correctness of our approach relies on the fact that the attackers have limited resources. If, in the example above, Eve is allowed to use extra hardware, then she can add up to K nontunable

receivers (one for each data channel). Now she can use her original tunable receiver for the challenges that are presented to her, while at the same time eavesdropping on all K channels using the nontunable receivers. Our protocol is not designed to counter such a committed hardware-oriented attack. We are assuming that Eve does not change the physical configuration of the system, yet she may decide to listen to someone else's conversation with the resources that she already has. A setting in which such conditions exist is, for example, a computer lab in an university. If there are surveillance cameras, this would deter users from changing the hardware configurations. However, it might still be possible for a user to modify the software in order to eavesdrop on the messages on the local network.

In this study, we will investigate a challenge-response approach to providing privacy in an optical broadcast network. Our primary measures in analyzing this approach will be the level of security provided, as well as the cost of security in terms of both the additional hardware and bandwidth required to implement the scheme and the degradation of network performance. The remainder of this paper is organized as follows. In Section 2, we define in detail our assumptions about the channels and the underlying scheduling protocol. In Section 3, we show how to modify any existing scheduling protocol so that the new protocol can detect eavesdropping. We present two solutions which guarantee privacy against attacks by several attackers working individually, and several attackers working in collaboration, in Sections 4 and 5, respectively. In Section 5, we also describe a dynamic channel allocation scheme, which can reduce the channel overhead required by the challenge-response scheme. We conclude the paper and provide some directions for future work in Section 6.

2 Assumptions about the Scheduling Scheme

We would like to make as few assumptions about the original scheduling protocol as possible. Our approach can therefore be applied to any existing protocol for multi-channel broadcast WDM networks. The protocol may employ deterministic scheduling in which every user has a predetermined channel and time slots to transmit [6–8]. The protocol will be a modified form of simple time-division multiplexing on the K channels. Alternatively, the protocol may be reservation-based, such that users ''compete'' for time slots on which they transmit, and the bandwidth allocated to each user over a period of time depends on the amount of information the user has to transmit [6–12]. Normally, protocols of the second type use at least one extra control channel on which reservations are made and resolved. Additional transmitters and receivers may be required for this control channel.

We assume that the propagation delay and the processing time are negligible and we also require that the protocol is collision free (i.e., users know in advance that they are not chosen to transmit, thus they do not interfere with the users who will transmit). Scheduling of transmissions is resolved early enough so that all users have sufficient time to tune their receivers and transmitters to the correct channels. Every user has one receiver which is tunable and one transmitter, which may or may not be tunable. In addition, users have one or more transmitters and receivers that may be dedicated for scheduling on the control channel. The tunability of the receiver is essential since our approach forces idle users to tune out of the data channels. Therefore, we cannot apply our approach to a protocol in which users have only fixed-tuned receivers. Finally, the protocol employs time slots of equal length, which correspond to the transmission time of one packet, and transmissions on all channels are synchronized; we assume that B bits are transmitted per time slot in each channel.

We denote the number of users in the system by N and the number of data channels by K $(K \leq N)$. The scheduling scheme of the original protocol can be summarized by introducing two functions, reads and sends which specify for every time slot t whether a user is receiving or sending information and on which channel.

$$\text{reads}(i,t) = \begin{cases} k & \text{if user } i \text{ is scheduled to receive on channel } k\ (1 \leq k \leq K) \text{ at time } t \\ nil & \text{if user } i \text{ is not scheduled to receive at time } t \end{cases}$$

$$\text{sends}(i,t) = \begin{cases} k & \text{if user } i \text{ is scheduled to transmit on channel } k\ (1 \leq k \leq K) \text{ at time } t \\ nil & \text{if user } i \text{ is not scheduled to transmit at time } t \end{cases}$$

Note that the semantics of the function $\text{reads}(i,t)$ also captures multicasting. For example,

$$\text{reads}(i_1,t) = \text{reads}(i_2,t) = \cdots = \text{reads}(i_j,t) \neq nil$$

implies that users $i_1, \ldots, i_j$ are all scheduled to receive a data packet on the same channel at time t. For the rest of this paper, however, we will focus only on point-to-point, i.e., unicast communications.

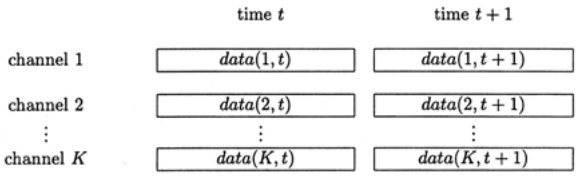

Fig. 1. The original protocol.

To further specify the operation of the protocol, we also define a function, data(k,t), which tells us what is the data transmitted on channel k at time t (see Fig. 1). It is useful to think about data(k,t) as a reference to a location (channel k and time t) from which data can be read and to which data can be written. This allows us to write data$(k,t)\leftarrow a$ to denote that the value of a is written to channel k at time t, and $b\leftarrow$data(k,t) to denote that the value of b is read from channel k at time t.

3 Secure Protocol

The main idea of the secure protocol is to exploit the tuning latency of the receivers in order to prevent eavesdropping. We want to guarantee that every idle user who is not receiving data is not tuned in to a data channel, since this would present an eavesdropping threat. We introduce a challenge channel and a response channel on which no sensitive data is transmitted. Every idle user is required to tune her receiver to the challenge channel. How can we guarantee that users comply? The execution of the protocol is supervised by a central authority, or principal. For every user, the principal sends a different challenge—a random string, over the challenge channel. In the next time slot, the principal expects every user to echo the same challenge, or a suitable function of the challenge, back on the response channel. Suppose that, at time $t+1$, for a given user, the principal receives a reply that is different from the challenge that was sent to the user at time t. This is a definite indication that the protocol was violated by the user if we assume error-free transmissions. Upon discovery of a discrepancy, the principal is authorized to take appropriate actions as discussed before.

On the other hand, if the principal receives the correct response, there are two possibilities. First, suppose that the user complies with the protocol: she tunes her receiver to the idle channel at time t, then she reads the challenge, and at time $t+1$ she echoes it back to the principal. The tuning latency guarantees that she cannot read from a data channel at time t: there is not enough time for her to tune in to the challenge channel, read the challenge, then retune to a data channel and read a non-negligible portion of the data there. This is, in fact, the normal mode of operation of the secure protocol.

However, there is a second possibility as seen in the following scenario. Suppose a user, Eve, is idle at time t, yet she tunes in to a data channel. At that time she cannot receive the challenge, since it is only available on the challenge channel. However, Eve may try to guess the challenge, and at time $t+1$ she can send back a random string, hoping that it coincides with the challenge. If Eve made a lucky guess, the principal cannot detect that Eve is in violation. However, the probability of her guessing a random string of b bits, is 2^{-b}. For example, even for a challenge as small as one byte $(b=8)$, Eve remains unnoticed with probability less than 0.4%. Thus, the probability of the principal missing a violation is negligible.

In the rest of this section, we will explain why we also need to modify the data channels and how we accomplish this. Our primary motivation for introducing challenge and response channels was to prevent idle users from eavesdropping on the data channels. However, we cannot assume that a user who is not idle (one who is scheduled to receive a message) follows the rules of the protocol either. This idea is illustrated in the following attack on the original protocol. Suppose Eve is scheduled to receive a very long message from Alice. At the beginning of the message Eve realizes that she already has a copy of the message and therefore she does not have to receive the remainder. At this time Eve can roam around the rest of the channels eavesdropping, while Alice continues to transmit. No other user, not even the principal, can notice that Eve is not listening on the scheduled channel since there is no feedback expected from her. In the worst case Eve may tune in to another channel and listen to a message from Carol to Dan without their consent.

To prevent the attack described above, the principal sends challenges not only to the idle users but also to the ones that are scheduled to receive on a data channel. The latter must read the data as well as the challenge on the same channel. Also, users who are scheduled to transmit in a slot will piggyback their response with the data. More precisely, we define

$D_k(t)$, the slot on the k-th data channel $(1 \leq k \leq K)$ at time t, to contain three fields:

$D_k(d,t)$	$D_k(c,t)$	$D_k(r,t)$

- **data**, $D_k(d,t)$
  This field corresponds to the data transmitted in the original protocol, that is,

  $$D_k(d,t) = \text{data}(k,t)$$

At time t, user i, for whom $k = \text{sends}(i,t)$, writes the data to channel D_k and user j, for whom $k = \text{reads}(j,t)$, receives the data from channel D_k.

- **challenge**, $D_k(c,t)$
  The principal writes in this field a challenge (a random string). User j who is supposed to tune in to channel D_k at time t in order to read data (that is, $k = \text{reads}(j,t)$) is also responsible for reading the challenge by the principal. This challenge is used in the next time slot, $t+1$, to verify that user j was indeed tuned to the proper channel.
- **response**, $D_k(r,t)$
  In this field, user i who is supposed to send data at time t on channel D_k (that is, $k = \text{sends}(i,t)$) writes her response to the principal, namely, the challenge that she received at time $t-1$. The principal reads the response and compares it to the challenge to verify it.

The three parts of the slot are bit- or byte-interleaved in order to prevent a user switching back and forth between a data and response channel within the same time slot. (The level of interleaving is a design consideration that will depend on the level of synchronization that can be achieved in the network.)

4 Solution against Attacks by Eavesdroppers Working Alone

In this section, we provide a solution that is secure against one or more eavesdroppers who are working alone. From now on, we will refer to this solution as Solution 1. However, in Section 5, we will show that Solution 1 is vulnerable to an attack by several collaborating eavesdroppers. We will then develop an improved version to handle such an attack.

4.1 Configuration of the Channels in Solution 1

We have modified the data channels as described in the previous section. Along with the K data channels $D_1, D_2, \ldots, D_K$, we have added one challenge channel, C, and one response channel, R (see Fig. 2). The challenge channel consists of time slots of equal length. Every time slot $C(t)$ is itself divided into N equal parts, with each part equal to the size of the challenge from the principal. We can think of $C(t)$ as an array indexed by the number of each user. We denote the i-th part of channel C at time t as $C(i,t)$; thus the slot $C(t)$, comprising N challenges on C may be written as follows:

$C(1,t)$	$C(2,t)$	$\ldots$	$C(N,t)$

Only the principal writes on the challenge channel. Every time slot of the challenge channel is bit interleaved.

The response channel has an identical structure. We denote the i-th part of channel R at time t as $R(i,t)$;

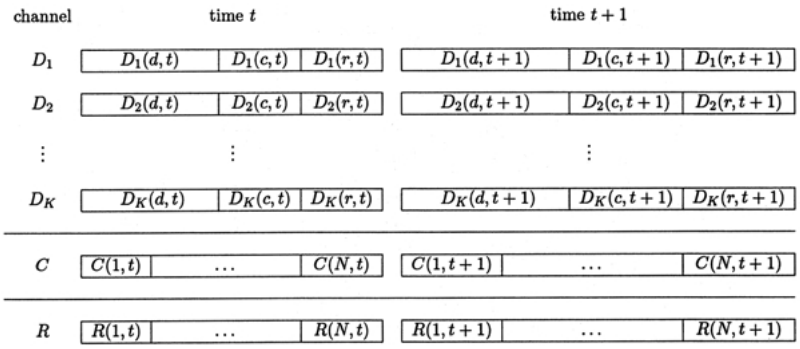

Fig. 2. The channels in Solution 1 (individual eavesdroppers).

thus the slot $R(t)$, comprising N responses on R may be written as follows:

$$R(1,t) \mid R(2,t) \mid \ldots \mid R(N,t)$$

Every user has one tunable receiver and one tunable transmitter. A slight modification of the protocol can be made to accommodate fixed-tuned transmitters. The principal must receive on and transmit to all the data channels. Also, the principal has to transmit to the challenge channel and receive on the response channels. This implies that the principal should have $K+1$ fixed-tuned receivers and $K+1$ fixed-tuned transmitters. These transmitters and receivers, as well as the additional challenge and response channels, are the primary fixed cost incurred by our modified approach.

4.2 The Protocol

How does the protocol work? At time t, every user has a specific location, including a channel and a field in the time slot, on which to transmit, and another specific location on which to receive. Users who are not scheduled to receive on any data channel must receive (challenges) on the challenge channel. Users who are not scheduled to transmit on any data channel must transmit (responses) on the response channel. At time t, the principal sends random bits as challenges to each user on the channel on which the user's receiver should be tuned. The principal expects the user to respond at time $t+1$ with the same sequence of random bits that the user received at time t.

First, we point out that the secure protocol does not alter the scheduling of data transmissions as defined in the original protocol. That is

$$D_k(d,t) = \text{data}(k,t).$$

On the other hand, we define

$$\text{challenge}(i,t) = \begin{cases} D_k(c,t) & \text{if reads}(i,t) = k \\ C(i,t) & \text{if reads}(i,t) = \text{nil} \end{cases}$$

to be the location at which user i expects a challenge at time t. If the user is scheduled to receive data, the challenge is located in the corresponding field, $D_k(c,t)$, in the scheduled data channel, D_k. If the user is not scheduled to receive data, then she has to listen to the challenge in the corresponding location, $C(i,t)$ on the challenge channel C.

We also define

$$\text{response}(i,t) = \begin{cases} D_k(r,t) & \text{if sends}(i,t) = k \\ R(i,t) & \text{if sends}(i,t) = \text{nil} \end{cases}$$

as the location in which a user is supposed to send her response. Note that, just like the challenges, the responses may be either on one of the data channels or on the response channel, depending on whether or not the user is scheduled to transmit data at that time.

The actions for user i $(1 \le i \le N)$ at time t are described in Fig. 3. In the two conditional statements, the user commits her transmitter to a specific channel and her receiver to a specific channel, which makes switching back and forth impossible.

The actions of the principal at time t are described in Fig. 4. The principal ensures the main property of the protocol, that is, for every user, i, and for every time slot, t,

$$\text{response}(i,t+1) = \text{challenge}(i,t).$$

If this property is violated for some pair (i,t), the principal can detect the violation by the end of time slot $t+1$. At time $t+2$, the principal already knows that user i is not complying with the protocol and appropriate actions can be taken.

4.3 Analysis of Solution 1

The protocol is successful against an attack by a single eavesdropper. At time slot t, Eve must commit to tuning her receiver to a single channel due to our

```
User(i,t)
// The procedure describes the actions of user i at time t
begin
  static 0
  // Assume that 0 stores the value of the
  // challenge from time t-1
  s ← sends(i,t)
  if (s ≠ nil)
    then
      tune transmitter to channel D_s
      D_s(d,t) ← data(s,t)
      D_s(r,t) ← 0
    else
      tune transmitter to channel R
      R(i,t) ← 0
  r ← reads(i,t)
  if (r ≠ nil)
    then
      tune receiver to channel D_r
      data(r,t) ← D_r(d,t)
      0 ← D_r(c,t)
    else
      tune receiver to channel C
      0 ← C(i,t)
end
```

Fig. 3. Actions of a user in Solution 1.

```
Principal(t)
//The procedure describes the actions of the principal at time t > 0.
//At time t=0, the principal sends challenges but does not check responses
begin
  O[N] // This array stores the old challenges
  for i = 1 to N do
    if response(i,t) ≠ O[i]
      then there is a problem with node i
  for i = 1 to N do
    O[i] ← random number
    challenge(i,t) ← O[i]
end
```

Fig. 4. Actions of the principal in Solution 1 and Solution 2.

assumption about the tunability delay. If Eve is idle (reads$(Eve, t) = \text{nil}$), then there are two choices for Eve. First, if Eve tunes her receiver to channel C, then she cannot eavesdrop. Second, if Eve decides to tune to a data channel, then she is not able to receive the challenge sent to her by the principal. Therefore, at time $t + 1$ she has to guess a response, and the chance of her remaining undetected after time $t + 1$ is 2^{-b} for a challenge of length b bits. A similar analysis applies in the case when Eve is scheduled to receive on one of the data channels, D_i, which has her bit-interleaved challenge, and she decides to eavesdrop on a different data channel, D_j $(i \neq j)$.

The selection of b depends on the level of security desired, with a longer challenge length providing a higher degree of security at a cost of reduced throughput. For Solution 1, the maximum challenge length is limited to $b_{1,\max} = B/N$, where B, as defined earlier, is the total number of bits per slot per channel.

To compute the throughput for Solution 1, we assume a normalized capacity of 1 per channel per slot. The data, challenge, and response channels then have a total capacity of $K + 1 + 1 = K + 2$. However, data is transmitted only on the K data channels, each having a throughput of $\frac{B-2b}{B}$. (On every data channel D_i, the challenge and response fields, $D_i(c, t)$ and $D_i(r, t)$, are not used for data. Each field has a length of b bits.) Overall, the throughput T_1 is given by:

$$T_1 = \frac{K}{(K+2)} \frac{B-2b}{B}.$$

If we want to achieve the maximum level of security, we can set $b = b_{1,\max} = B/N$, yielding the following throughput:

$$T_1 = \frac{K}{(K+2)}(1 - 2/N) = \frac{K}{(K+2)} \frac{(N-2)}{N}.$$

Thus, in the extreme case when all N users have to share a single data channel, that is, $K = 1$, the throughput is just 1/3 for $N \gg 1$. On the other hand, for larger values of K we obtain a throughput close to 1 (see Fig. 5), which means that, in Solution 1, we have added security without noticeably depreciating the throughput performance of the underlying scheduling protocol. Note that asymptotically the throughput of Solution 1 is close to 1, which suggests that the approach is scalable in the number of users, N, and the number of channels, K; however, if N is large compared to B, then the challenge length will be small, resulting in a lower level of security.

It is interesting to note that Solution 1 is secure against a simultaneous attack by any number of non-collaborating users. However, it is vulnerable to an attack by two or more collaborating eavesdroppers. Suppose that there are several users who have decided to cooperate in order to allow one of them to eavesdrop. For simplicity, we assume that there are just two collaborators: Eve and Eric. Eve tunes in to her favorite data channel and starts to eavesdrop. Normally, she would be detected within two time slots, but now Eric covers for her. If Eric and Eve are both idle, then their challenges are received on C and they have to respond on R. However, Eric can read both the challenge directed to him and the challenge addressed to Eve. Furthermore, Eric can respond for himself and for Eve. The principal has no means of detecting that someone else has served as a proxy for

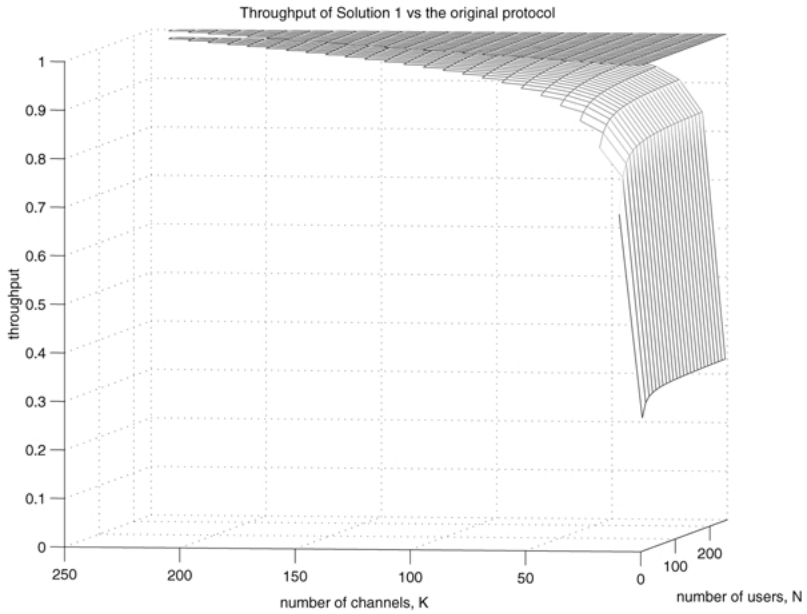

Fig. 5. Comparison of the throughputs of the original protocol (above) and Solution 1 (below).

Eve, and her violation remains unnoticed. A solution to this problem will be addressed in Section 5.

4.4 Application of Solution 1 to an Existing Protocol

We now illustrate how Solution 1 may be applied to an existing media access control protocol. We choose the dynamic time-wavelength division multiple access (DT-WDMA) protocol described in Chen et al. [9].

In DT-WDMA, nodes are connected to a passive-star coupler which is an optical broadcast medium. The system requires a single control channel and N data channels, where N is the number of nodes. Each node is assigned its own unique channel on which to transmit data packets, but may receive data packets on any of the data channels. Thus, each node requires a fixed-tuned transmitter and a fixed-tuned receiver to access the control channel, a fixed-tuned transmitter to access a single data channel, and a tunable receiver to access all data channels. Time is divided into slots, with a slot length equal to the packet transmission time plus the receiver tuning time. The control channel is accessed using a TDM-based approach. Each slot on the control channel contains N minislots, one for each of the N nodes in the network. When a node i has a packet to transmit, it will send a request in its assigned minislot. The request indicates the source node, the destination node, and a priority level. After a node transmits its request in a minislot in time slot t, it will transmit its data packet in the following time slot $(\text{sends}(i, t+1) = i)$ on its assigned channel. Since each node continuously monitors the control channel, a node will be able to determine whether it will receive a packet in the following time slot and the channel to which it should tune its receiver. If node j receives a request from node i at time t, it will set $\text{reads}(j, t+1) = i$ and tune its receiver to channel i in slot $t+1$. If more than one node transmits a request on the control channel to the same destination node, then the destination node will receive the packet with the highest priority. If the priorities are equal, then the destination node will choose one of the transmissions based on some deterministic algorithm, such as choosing the node with the lowest index.

Modifying this protocol to accommodate Solution 1 requires one additional channel to serve as a challenge channel and one additional node to serve as the principal. The principal node requires $N+1$ fixed-tuned transmitters for the data channels and the challenge channel, and $N+1$ fixed-tuned receivers for the data channels and the control channel. Since the principal monitors the control channel, it will know on which channel each of the other nodes' receiver should be tuned in a given time slot. The

principal can then transmit the challenge to the node on the appropriate channel. If a node is not scheduled to receive a packet in a given time slot, then its receiver should be tuned to the challenge channel. Each node will respond to the challenge in the following time slot on the data channel on which its transmitter is fixed. Note that, for this particular protocol, there does not exist a problem of collaborating eavesdroppers. Since each node must respond to the challenge on its own channel, and since each node only has a single fixed-tuned transmitter for the data channels, one node cannot respond to a challenge for another node.

5 Solution against Attack by Two or More Collaborating Eavesdroppers

In this section, we describe Solution 2, which is a more elaborate version of the secure protocol. This protocol prevents an attack by two or more collaborating eavesdroppers as was described in Subsection 4.3. The main approach for improving Solution 1 is to increase the number of challenge and response channels, and to assign each user a distinct challenge-channel/response-channel pair. Since no two users share both the same challenge channel and the same response channel, one user will not be able to respond to another user's challenge while also responding to his or her own challenge.

5.1 Configuration of the Channels in Solution 2

We assume that the number of users, N, is a square of an integer, let $M^2 = N$. This can be done without loss of generality: if N is not a square, we can pick $M = \lceil \sqrt{N} \rceil$. The K data channels $D_1, D_2, \ldots, D_K$ are defined exactly as in Solution 1. However, now instead of a single pair of challenge-response channels, there are M challenge and M response channels, denoted as $C_1, C_2, \ldots, C_M$ and $R_1, R_2, \ldots, R_M$, respectively. As shown in Fig. 6, every challenge channel C_i $(1 \leq i \leq M)$ is divided into time slots of equal length. A time slot $C_i(t)$ on channel C_i is itself subdivided into M equal parts. Similar to Solution 1, we can think of $C_i(t)$ as an array indexed from 1 to M.

$$C_i(t) = \boxed{C_i(1,t) \mid C_i(2,t) \mid \ldots \mid C_i(M,t)}$$

$C_i(j,t)$ is the j-th part of $C_i(t)$ and it represents a single challenge sent by the principal. As in Solution 1, this array representation is only conceptual, because in reality the bits of all fields of $C_i(t)$ are interleaved. The response channels are divided in an identical way, so that $R_i(t)(1 \leq i \leq M)$ denotes the slot of response channel R_i at time t, consisting of M parts:

$$R_i(t) = \boxed{R_i(1,t) \mid R_i(2,t) \mid \ldots \mid R_i(M,t)}$$

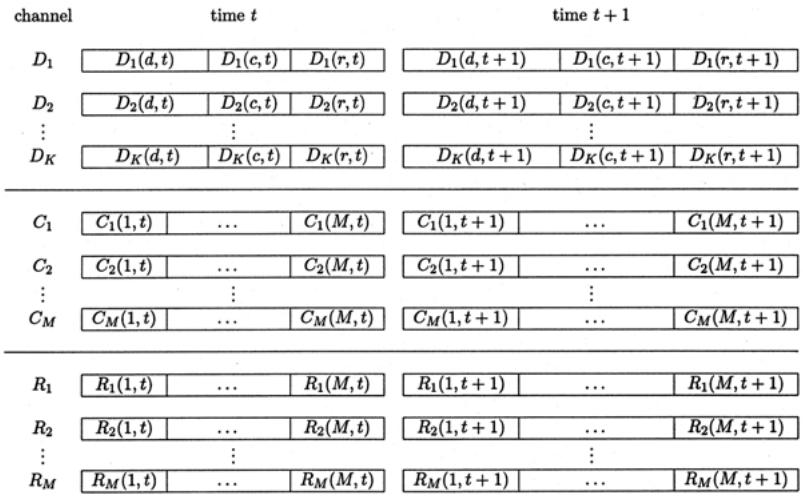

Fig. 6. The channels in Solution 2 (collaborating eavesdroppers).

$R_i(j,t)$ is the j-th part of $R_i(t)$ and represents a single response sent to the principal.

Every user has a tunable receiver and a tunable transmitter. The principal needs $K+M$ fixed-tuned transmitters and $K+M$ fixed-tuned receivers.

5.2 The Protocol

As already mentioned, the data fields are defined exactly as in Solution 1. To complete the description of Solution 2, it suffices to redefine the semantics of the functions challenge and response.

For a fixed M, let us arrange the users as entries in a $M \times M$ matrix, in row-major order, such that every user is in a distinct row and distinct column of the matrix. For example, if $N = 9$ and $M = 3$, the matrix looks like this

$$\begin{matrix} 1 & 2 & 3 \\ 4 & 5 & 6. \\ 7 & 8 & 9 \end{matrix}$$

We introduce functions, which for any user i, $(1 \le i \le N)$, help us to address the user as an entry in the $M \times M$ matrix. For $1 \le a \le M$ and $1 \le b \le M$ define

$$h(a,b) = M(a-1)+b \quad f(h(a,b)) = a \quad g(h(a,b)) = b.$$

Given row a and column b in the matrix, $h(a,b)$ returns the index of the user corresponding to that entry. Also, for user i, $f(i)$ and $g(i)$ compute the corresponding row and column indices, respectively. In the example above, $h(3,2) = 8$, $f(7) = 3$, and $g(6) = 3$.

We denote by challenge(i,t) the location at which user i expects a challenge at time t. If the user is scheduled to receive data, then the challenge is located in the corresponding field of the scheduled data channel. If the user is not scheduled to receive data, then the challenge is in part $g(i)$ of the appropriate challenge channel, $C_{f(i)}$.

$$\text{challenge}(i,t) = \begin{cases} D_k(c,t) & \text{if reads}(i,t) = k \\ C_{f(i)}(g(i),t) & \text{if reads}(i,t) = \text{nil}. \end{cases}$$

We also define response(i,t) as the location at which user i is supposed to send her response. If the user is scheduled to transmit data, then the response is located in the corresponding field of the scheduled data channel. If the user is not scheduled to transmit data, then her response is in part $f(i)$ of the appropriate response channel, $R_{g(i)}$.

```
User(i,t)
// The procedure describes the actions of user i at time t
begin
  static 0
  // Assume that 0 stores the value of the
  // challenge from time t-1
  s ← sends(i,t)
  if (s ≠ nil)
    then
      tune transmitter to channel D_s
      D_s(d,t) ← data(s,t)
      D_s(r,t) ← 0
    else
      tune transmitter to channel R_g(i)
      R_g(i)(f(i),t) ← 0
  r ← reads(i,t)
  if (r ≠ nil)
    then
      tune receiver to channel D_r
      data(r,t) ← D_r(d,t)
      0 ← D_r(c,t)
    else
      tune receiver to channel C_f(i)
      0 ← C_f(i)(g(i),t)
end
```

Fig. 7. Actions of a user in Solution 2.

$$\text{response}(i,t) = \begin{cases} D_k(r,t) & \text{if sends}(i,t) = k \\ R_{g(i)}(f(i),t) & \text{if sends}(i,t) = \text{nil}. \end{cases}$$

Note that, if user i is neither scheduled to receive nor scheduled to transmit data, the challenge is at location $C_{f(i)}(g(i),t)$, while the response is at location $R_{g(i)}(f(i),t)$. Since for every i and j, such that $1 \le i < j \le M^2$ we have $f(i) \ne f(j)$ and/or $g(i) \ne g(j)$, then it follows that no two users i and j can both share the same challenge and the same response channels: therefore, no user can cover for another.

The actions of every user i, where $1 \le i \le N$, are described in Fig. 7. The actions of the principal are described in Fig. 4. Note that the principals in Solution 1 and Solution 2 have different definitions of the functions challenge and response; therefore, the two principals perform different actions even though their code looks identical.

5.3 Analysis of Solution 2

The key observation is that Solution 2 guarantees that no two users expect challenges and send back responses on the same pair of channels. We can show that, if Eve and Eric want to collaborate and eavesdrop, then their attempt will be detected. Without loss of generality, assume that Eve eavesdrops and Eric tries to send back a correct response on her behalf. At least one of the following is true: Eric is assigned a challenge channel different from Eve's, or

Eric is assigned a response channel different from Eve's.

First, if Eric's challenge channel is different from Eve's, then he can read only one of the challenges and must guess the second challenge. With high probability his violation is detected in the next time slot; in fact, this probability is equal to $2^{-(B/M)}$. Second, if Eric's response channel is different from Eve's, then he cannot send back to the principal both his and Eve's response. Again, the violation is detected. The argument above can be generalized by induction to any number of collaborating eavesdroppers.

We expect the throughput of Solution 2 to be less than that of Solution 1 since we had to prevent attacks by collaborating eavesdroppers. For this solution, we had to increase the number of challenge and response channels—channels on which no data is sent. To compute the throughput, let $M = \sqrt{N}$ be the number of users that share the same challenge or response channel. If we assume a capacity of 1 for a channel, then the entire system, including all the data, challenge, and response channels, has a capacity of $K + M + M = K + 2M$. However, the K data channels only have a throughput of $\frac{B-2b}{B}$. Recall that on each data channel D_i, we have the challenge and response fields, $D_i(c,t)$ and $D_i(r,t)$, each of which has length b bits. For the throughput, T_2 we obtain

$$T_2 = \frac{K}{(K + 2M)} \frac{B - 2b}{B}.$$

For the maximum level of security, we set b equal to $b_{2,\max} = \frac{B}{M}$, which yields the throughput:

$$T_2 = \frac{K}{(K + 2M)}(1 - 2/M) = \frac{K}{(K + 2M)} \frac{(M - 2)}{M}.$$

This throughput is is approximately $c/(c+2)$ for $K = c\sqrt{N}$ and $M \gg 1$. Fig. 8 depicts the drop in throughput of Solution 2 in comparison with the throughput of Solution 1.

5.4 Improving the throughput of Solution 2

In this subsection, we present two modifications to Solution 2. Just like Solution 2, the new protocols are secure against collaborating eavesdroppers, yet they also have better throughput and use fewer additional (non-data) channels.

5.4.1 Modified Solution 2

Note that for all i, $(1 \le i \le \sqrt{N})$, we can combine the challenge channel C_i and the response channel R_i into a single challenge-response channel CR_i. With all other things being equal, this will decrease the number of additional channels in Solution 2 from $2\sqrt{N}$ to $\sqrt{N}$. The drawback is that we also decrease by half the

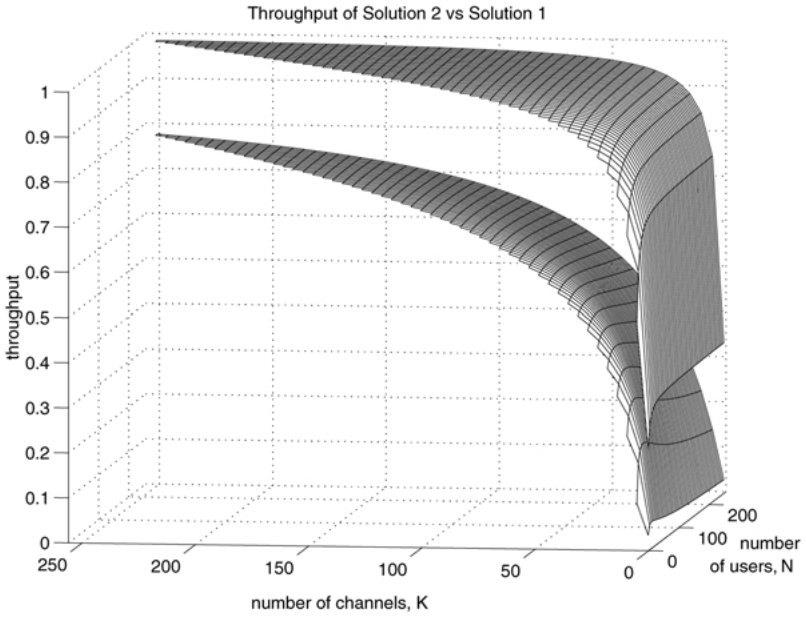

Fig. 8. Comparison of the throughputs of Solution 1 (above) and Solution 2 (below).

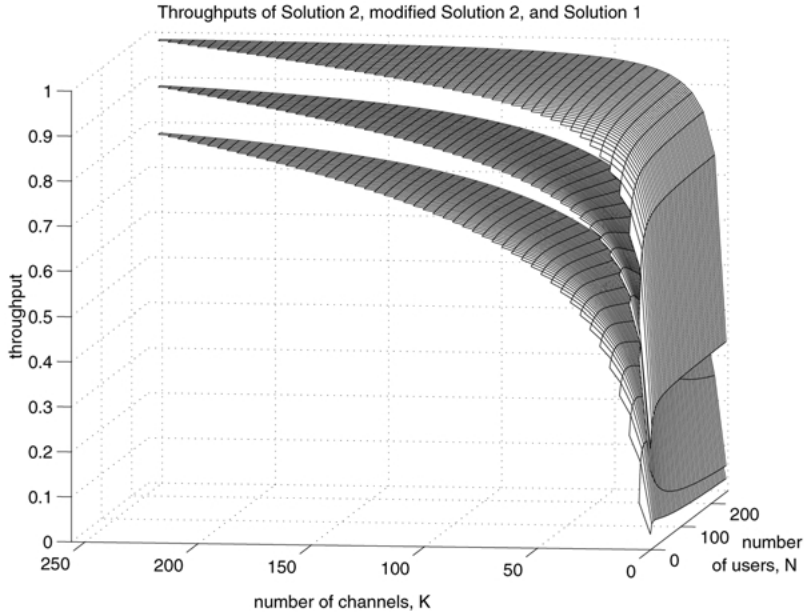

Fig. 9. Comparison of the throughputs of Solution 1 (above), Solution 2 (below), and the Modified Solution 2 (middle).

maximum challenge length, thus increasing the chance for a correct guess of a response. However, if the original challenge was B/M bits, the size of the new one will be $B/(2M)$; therefore, the probability that a guess will be unnoticed is just $2^{-B/(2M)}$, which is still sufficiently low for reasonably large B. The throughput of the modified Solution 2 is

$$T'_2 = \frac{K}{(K+M)}\frac{B-2b}{B}.$$

With the maximum challenge length of $b'_{2,\max} = \frac{B}{2M}$, we obtain

$$T'_2 = \frac{K}{(K+M)}(1-1/M) = \frac{K}{(K+M)}\frac{(M-1)}{M}$$

which is much closer to the throughput of Solution 1. The comparison of throughputs for the different solutions is shown in Fig. 9. The upper surface represents the throughput of Solution 1, T_1. The lower surface represents the throughput of Solution 2, T_2. Finally, the throughput of the modified Solution 2, T'_2, is in the middle. Clearly, there is a trade-off between the throughput of a solution and the probability of detecting an eavesdropper.

5.4.2 Dynamic Channel Allocation

In the previous protocols, users are assigned to challenge and response channels in a static manner. We note that, when a user is transmitting or receiving data, the user does not utilize his assigned challenge or response channels. On the other hand, the only time that the challenge and response channels will be fully utilized is when none of the users are transmitting or receiving on the data channels, in which case, all of the data channels will be idle.

It is possible to reduce the number of dedicated challenge/response channels by dynamically assigning a channel as either a challenge/response channel or a data channel depending on how many users are transmitting and receiving data and how many users are idle. If a data channel is not being used to transmit data, it may be used instead as a challenge/response channel.

In order to implement a dynamic channel allocation scheme, the previous algorithms must be modified such that, in each time slot, the data channels and the challenge/response channels are assigned based on the number of busy and idle users. The channel allocation can either be performed in a distributed manner or by the principal. For a given time slot, each user will indicate over the control channel whether or not they have data to transmit in that time slot. The channel

allocation algorithm will then assign data channels to those users who are transmitting or receiving data in the time slot. The remaining users will then be dynamically assigned challenge/response channels in a similar way as in Section 5.2. Since challenge/response channels only need to be allocated for the idle users, the total number of challenge/response channels required will be the square root of the number of idle users. Furthermore, data channels that are idle in the specified time slot can be reallocated as challenge/response channels, reducing the need for additional dedicated challenge/response channels.

As an example, consider the case in which there are $N = 16$ users and $K = 12$ data channels. If 12 of the 16 users are transmitting and receiving on the data channels, then there will be four idle users who must be confined to the challenge/response channels. In this case, only $\sqrt{4} = 2$ additional channels are required as challenge/response channels. If only seven of the 16 users are transmitting and receiving on the data channels, then there will be nine idle users requiring $\sqrt{9} = 3$ challenge/response channels. However, since five of the data channels will be idle, these data channels can be used as the challenge/response channels, and no additional channels will be required.

We now calculate the total number of channels required for a system with N users and at most K data channels. Let L be the total number of challenge/response channels required in addition to the K data channels. Also, let n be the number of users out of N users that are transmitting and receiving data, with $n \leq K \leq N$. If n users are on the data channels, then $K - n$ of the data channels will be idle. The total number of channels available as channel/response channels is then $K - n + L$. These available channels can accommodate up to $(K - n + L)^2$ idle users, ensuring that no two idle users share both the same challenge channel and the same response channel. The total number of idle users is $N - n$; thus, as long as $(K - n + L)^2 \geq N - n$, there will be a sufficient number of channels. Solving for L, we obtain the following requirement for the number of additional channels:

$$L = \max_n \left(\sqrt{N - n} - K + n \right).$$

Since the above equation is strictly increasing with n, the worst case occurs when $n = K$. The requirement for L is then:

$$L = \lceil \sqrt{N - K} \rceil,$$

and the total number of channels required in the system is $K + \lceil \sqrt{N - K} \rceil$.

We now compute T_d, the maximum overall throughput for the dynamic challenge-response solution:

$$T_d = \frac{K}{(K + L)} \frac{(B - 2b)}{B}.$$

The maximum possible challenge length is $b_{d,\max} = \frac{B}{2L}$. Thus, the throughput per data channel is $(L - 1)/L$. For the overall throughput we have:

$$T_d = \frac{K}{(K + L)} \frac{(L - 1)}{L}.$$

We also note that the probability that a guess at a challenge will go unnoticed is $2^{-B/(2L)}$.

Fig. 10 compares the throughputs of Solution 2, Modified Solution 2 and the dynamic challenge allocation (DCA) solution. Note that there is a trade-off between throughput, security and number of additional channels.

Table 1 compares the overhead and performance penalties incurred by the different solutions and the probability of compromising each of them by randomly guessing the challenge. We assume that the throughput of the original protocol is 1. From the table, it is is clear that Solution 1 has the lowest overhead in terms of the number of challenge/response channels, and the decrease in throughput is also minimized. However, the solution only works for eavesdroppers working independently.

From Table 1 we see that the DCA solution offers better security than Modified Solution 2, since $M > L$, and therefore more bits $(B/2L)$ are allocated in the challenge field for the DCA solution; however, the former has slightly less throughput as seen in Table 1 and Fig. 10. We also note that the DCA solution requires fewer overhead channels than Solution 2 or Modified Solution 2.

In Figs 11 and 12, we compare all the schemes based on the probability that a guess can equal the challenge. These curves were generated using the formulae listed in Table 1. We consider a slot length of $B = 300$ bits. From both figures, we see that Solution 1 has the worst performance, and Modified Solution 2 has the next-worst performance on the security metric. From Fig. 11, we observe that if the number of channels is significantly lower than the number of

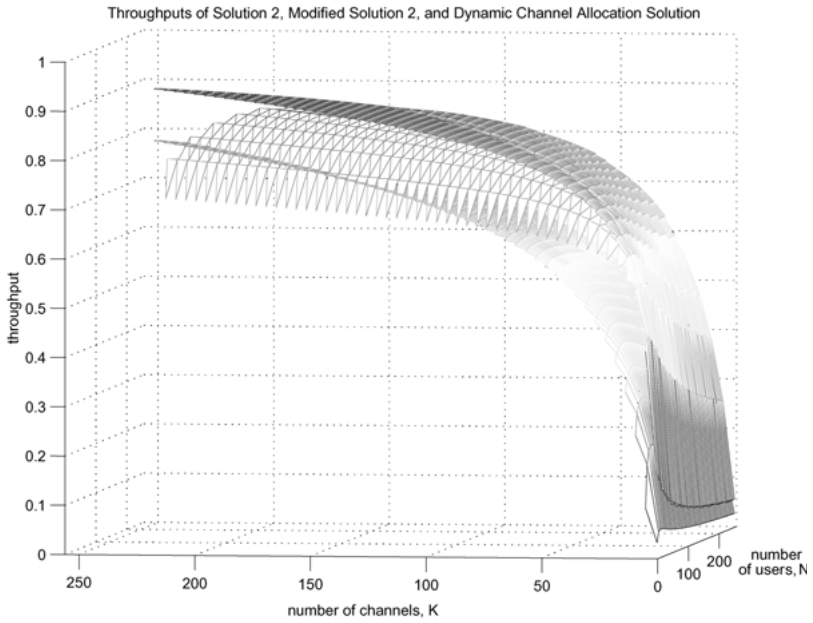

Fig. 10. Comparison of the throughputs of Solution 2 (below), Modified Solution 2 (above) and dynamic channel allocation solution (middle).

users, the security performance of DCA is close to that of Modified Solution 2, and Solution 2 has the best level of security. As the number of channels approaches the number of users, the performance of DCA surpasses that of Solution 2. These observations are confirmed in 12.

We also note that within the various versions of Solution 2, there is some flexibility in picking the actual length of the challenge. Thus the throughput can be traded for a higher level of security. If we assume that the challenge length in all three of the protocols is held fixed to some constant, i.e., if the probability of detecting violations is held constant for these protocols, then it is fairly easy to show that the DCA Solution has better throughput than Modified Solution 2. This trade-off is illustrated in Fig. 13. Each curve is obtained by varying the challenge length from $b = 1$ to $b_{\max}$ for each of the schemes, and plotting the corresponding probability of guessing a challenge versus the throughput.

5.5 Application of Solution 2 to an Existing Protocol

Let us now apply Solution 2 to an existing media access protocol. Consider a variation of the DT-WDMA protocol in which the number of data channels, K, is less than the number of users, N. Each node is equipped with a tunable transmitter rather than a fixed-tuned transmitter to access the data channels. The format of the control channel remains the same. A request message, in addition to the source, destination, and priority information, also specifies

Table 1. Performance comparison of the various challenge-responses solutions. N = number of users in the system; K = number of data channels $(1 \le K \le N)$; $M = \lceil \sqrt{N} \rceil$; $L = \lceil \sqrt{N-K} \rceil$; B = number of bits in a data slot.

Protocol	No. of Challenge/Response Channels Needed (Overhead)	Throughput	Probability of Correctly Guessing a Challenge
Solution 1	2	$T_1 = \frac{K}{K+2} \cdot \frac{N-2}{N}$	$2^{-\lfloor \frac{B}{N} \rfloor}$
Solution 2	$2M = 2\lceil \sqrt{N} \rceil$	$T_2 = \frac{K}{K+2M} \cdot \frac{M-2}{M}$	$2^{-\lfloor \frac{B}{M} \rfloor}$
Modified Solution 2	$M = \lceil \sqrt{N} \rceil$	$T'_2 = \frac{K}{K+M} \cdot \frac{M-1}{M}$	$2^{-\lfloor \frac{B}{2M} \rfloor}$
DCA Solution	$L = \lceil \sqrt{N-K} \rceil$	$T_d = \frac{K}{K+L} \cdot \frac{L-1}{L}$	$2^{-\lfloor \frac{B}{L} \rfloor}$

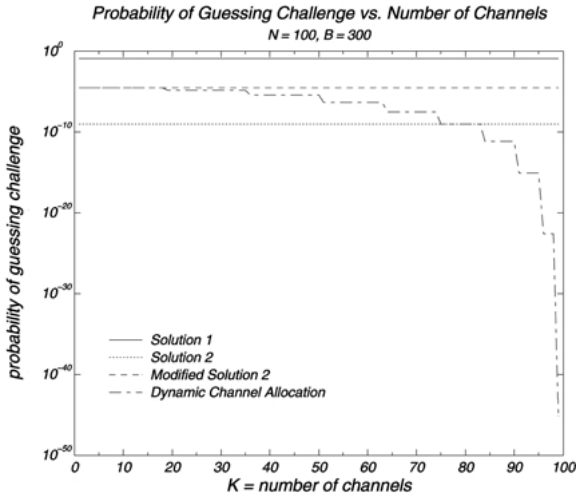

Fig. 11. Probability of guessing a challenge versus K for various solutions.

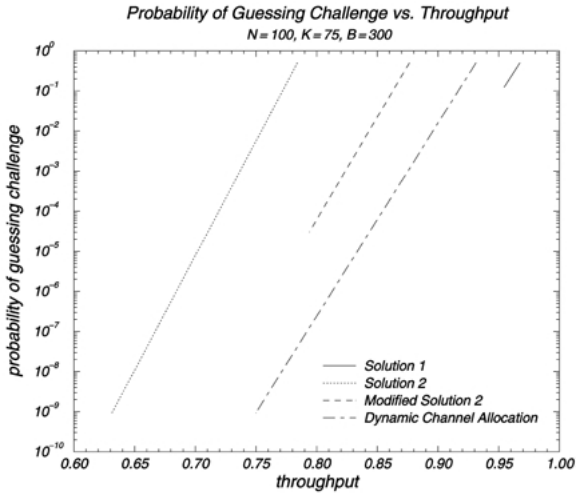

Fig. 13. Probability of guessing a challenge versus throughput for varying challenge length.

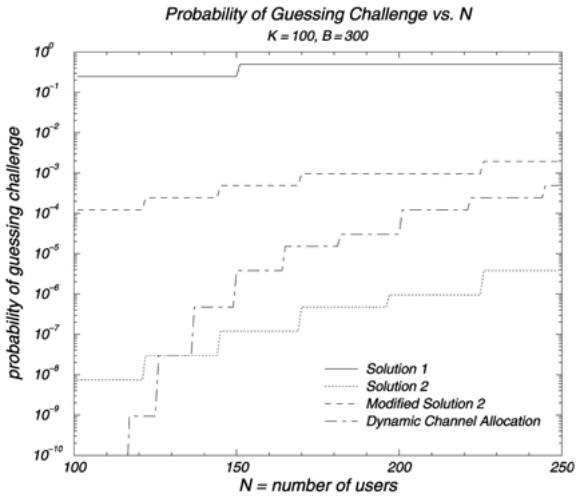

Fig. 12. Probability of guessing a challenge versus N for various solutions.

the channel on which the transmission is to take place. If more than one node request to transmit on the same channel in a given slot, the request with the highest priority is allowed to transmit. A deterministic algorithm can be used to break further ties.

In order to apply Solution 2 to this protocol, we require an additional $2\sqrt{N}$ channels. Half of these channels will be allocated as challenge channels, while the other half will be allocated as response channels. Each node is assigned one challenge channel and one response channel in such a way that no node shares the same challenge-channel/response-channel pair. We also need to add a principal which is equipped with $K + \sqrt{N}$ fixed-tuned transmitters for the data and challenge channels and $K + \sqrt{N} + 1$ fixed-tuned receivers for the data, response, and control channels. Since the principal only needs to listen to the control channel, it does not require an extra transmitter for the control channel. A node which is scheduled to transmit or receive a packet in a given time slot will tune its transmitter or receiver, respectively, to the appropriate channel. Nodes that are not scheduled to transmit must tune their transmitters to the assigned response channel, while nodes that are not scheduled to receive must tune their receivers to the assigned challenge channel. The principal will send a challenge to a node either on a data channel or on a challenge channel, depending on whether the node is scheduled to receive a packet or not. In the following time slot, the principal will expect a response on either a data channel or on a response channel, depending on whether or not the node is scheduled to transmit a data packet.

6 Conclusions and Future Work

In this paper, we have developed a new method for detecting and preventing unauthorized disclosure of information at the media access level in a multi-

channel network, such as a broadcast-and-select WDM optical network. Our approach is not restricted to a single protocol. Rather, it is a methodology which can be applied to any collision-free media access control protocol that uses tunable receivers. The approach presented does not use cryptography; instead, it relies on the resource limits of potential attackers and on a challenge-response scheme to guarantee that, at any time, idle users are not eavesdropping on the data channels. If a user does not comply with the protocol, there is a very high probability that the violation will be detected, and appropriate measures can be taken. We presented a solution for attacks by individual eavesdroppers as well as a solution for eavesdroppers working in collaboration, and illustrated how these solutions can be applied to existing protocols. For the latter approach (Solution 2) it was shown that, if dedicated challenge and response channels are used, then at least $2\sqrt{N}$ additional channels are required as challenge and response channels. It is further shown that, by appropriately modifying the scheme to allow the sharing of bandwidth by data channels and challenge/response channels, the total number of required channels can be reduced to $K + \lceil \sqrt{N - K} \rceil$.

Since, in the proposed scheme, an attacker is able to intercept at least one slot of data before any action can be taken, a possible approach for improving the scheme is to spread the contents of a private message over a number of slots [4, 3], such that the message is only compromised if more than a certain number of slots are intercepted. Investigation of such an improvement is a topic for future research. Further work is also required to define the appropriate action to take once a violation is detected. One possible action is to halt the transmission of sensitive information until compliance with the protocol has been detected over a number of slots. Algorithms need to be developed to determine how many slots a user should wait before resuming the transmission of private data.

Another interesting question that needs to be addressed is how the principal can obtain truly random bits for the challenges. Indeed, truly random bits are generated very slowly, thus they are impractical in real applications. Fortunately, there are very fast generators for pseudorandom bits. Although as of today no algorithm for pseudorandom bits is proven to be secure, the ones that are used in practice have good random properties which makes our method feasible. A second issue is whether, if we have a reliable source of pseudorandom bits, it is easier to implement cryptography using a method similar to Shannon's ''one-time pad'' [5]. However, such an approach would require the sender and the receiver to share the same sequence of random bits, that is, create the same sequence at two different locations. On the other hand, in our method the random bits are created in a single location by the principal and they do not have to be shared.

In Solution 2, we assumed that every idle user receives challenges and sends responses on a distinct pair of channels. It would be interesting to explore whether randomization of the challenge and response channels can provide privacy against collaborators while using fewer than $2\sqrt{N}$ additional channels, where N is the total number of users.

Although we consider our scheme and encryption to be complementary techniques for providing privacy, it is useful to compare the two approaches. Cryptography can provide a high level of privacy by making it difficult for an attacker to decrypt an encrypted message. However, conventional cryptography offers no satisfactory method to detect possible attacks on the privacy of a transmission and no means to determine that a message has been compromised. Also, as we have mentioned, cryptography requires additional computation time for encryption and decryption, which may not be tolerable for real-time applications, and key management may become cumbersome. On the other hand, our scheme provides a mechanism for detecting and identifying possible eavesdroppers, and the scheme can provide information as to whether or not data has been transmitted privately. The drawback is that the scheme may allow a limited amount of private data to reach the attacker before preventive measures can take place. Furthermore, the proposed scheme requires an additional trusted server to act as a principal, and the scheme also requires additional bandwidth for the challenge channels and the response channels. We see that both the proposed scheme and encryption have their own strengths and weaknesses. By combining these two schemes, not only can we achieve a high degree of privacy, but we can also identify and take actions against potential attackers.

A further direction of research may be the development of a hybrid protocol that can switch dynamically between two modes—one without

protection of privacy, and one in which privacy is guaranteed. A good solution would be able to use the maximum bandwidth in the unprotected mode, while upon request and suitable payment, it will have security in the private mode.

References

[1] B. Schneier, Applied Cryptography, second ed. (John Wiley & Sons, 1996).

[2] G. Brassard, Modern Cryptology (Springer-Verlag, 1988).

[3] I. Andonovic, L. Tančevski, Optical network architectures utilizing wavelength hopping/time spreading codes, International Journal of Optoelectronics, vol. 11, no. 1, (1997), pp. 1–9.

[4] L. Tančevski, I. Andonovic, Hybrid wavelength hopping/time spreading schemes for use in massive optical networks with increased security, IEEE/OSA Journal of Lightwave Technology, vol. 14, no. 12, (Dec. 1996), pp. 2636–2647.

[5] C. E. Shannon, Communication theory of secrecy systems, Bell System Technical Journal, vol. 28, no. 4, (Oct. 1949), pp. 656–715.

[6] I. Chlamtac, A. Ganz, Channel allocation protocols in frequency-time controled high speed networks, IEEE Transactions on Communications, vol. 36, no. 4, (Apr. 1988), pp. 430–440.

[7] A. Ganz, Y. Gao, A time-wavelength assignment algorithm for a WDM star network, in Proceedings, IEEE INFOCOM '92, (Florence, Italy), (May 1992), pp. 2144–2150.

[8] M. S. Borella, B. Mukherjee, Efficient scheduling of nonuniform packet traffic in a WDM/TDM local lightwave network with arbitrary transceiver tuning latencies, IEEE Journal on Selected Areas in Communication, vol. 14, no. 5, (June 1996), pp. 923–934.

[9] M.-S. Chen, N. R. Dono, R. Ramaswami, A media access protocol for packet-switched wavelength division multiaccess metropolitan area networks, IEEE Journal on Selected Areas in Communications, vol. 8, no. 6, (Aug. 1990), pp. 1048–1057.

[10] G. N. M. Sudhakar, N. Georganas, M. Kavehrad, Slotted ALOHA and reservation ALOHA protocols for very high-speed optical fiber local area networks using passive star topology, IEEE/OSA Journal of Lightwave Technology, vol. 9, no. 10, (Oct. 1991), pp. 1411–1422.

[11] B. Mukherjee, WDM-based local lightwave networks—Part I: Single-hop systems, IEEE Network Magazine, vol. 6, no. 3, (May 1992), pp. 12–27.

[12] F. Jia, B. Mukherjee, J. Iness, Scheduling variable-length messages in a single-hop multichannel local lightwave network, IEEE/ACM Transactions on Networking, vol. 3, no. 4, (Aug. 1995), pp. 477–488.

[13] P. P. Iannone, N. J. Frigo, K. C. Reichmann, Enhanced privacy in broadcast passive optical networks through use of spectral slicing in waveguide grating routers, in Proceedings, OFC '97, (Dallas, TX), (Feb. 1997), pp. 52–53.

Boris H. Simov studied informatics at St. Kliment Ohridski University, Sofia, Bulgaria, received a B.S. degree in Computer Science and Mathematics from Graceland College, Lamoni, IA in 1993, and M.S. degree in Computer Science from Iowa State University, Ames, IA in 1995. He is a Ph.D. candidate in Computer Science at Iowa State University. In May 2000 Mr. Simov became a member of the technical staff in the Enterprise Systems Technology Lab, Hewlett-Packard. His research interests are in computer security and in target tracking. He is a member of the ACM and IEEE.

Jason P. Jue received his B.S. degree in Electrical Engineering from the University of California, Berkeley in 1990, an M.S. degree in Electrical Engineering from the University of California, Los Angeles in 1991, and a Ph.D. degree in Computer Engineering from the University of California, Davis in 1999. In 1999 he joined the faculty at the University of Texas at Dallas, where he is currently an Assistant Professor of Computer Science and a member of the Center for Advanced Telecommunications Systems and Services. Dr. Jue is actively conducting research in the area of WDM optical networks, focusing on the design and analysis of optical network architectures and protocols. He served as the techincal program chair for OptiComm 2000, and is a contributing editor to Optical Networks Magazine.

Srini Tridandapani received his B.E. degree in Electronics and Communication Engineering from Anna University Madras, India in 1988, M.S. and Ph.D. degrees, both in Electrical Engineering from the University of Washington, Seattle in 1990 and 1994, respectively, and an M.D. degree from the University of Michigan, Ann Arbor in 2001.

During the academic year 1994–1995, Dr. Tridandapani was a post-doctoral researcher in Computer Science at the University of California, Davis, and during the academic years 1995–1997 he was on the faculty of Electrical and Computer Engineering at Iowa State University, Ames. He is currently a Research Investigator in the Department of Radiology at the University of Michigan, Ann Arbor and a Resident Physician at St. Joseph Mercy Health System in Ypsilanti, Michigan.

Dr. Tridandapani is interested in high-speed computer and communication systems, especially with their application to health care systems. More recently, he has been involved in bioengineering research focussed on ultrasound imaging systems.