Distributed Algorithms for Attack Localization in All-Optical Networks

Ruth Bergman Muriel Médard Serena Chan

Massachusetts Institute of Technology
Lincoln Laboratory
E-mail: ruth@ll.mit.edu, medard@ll.mit.edu, chans@mit.edu

Abstract

All-Optical Networks provide ultra-fast data rates, but present a new set of challenges for network security. We present a new algorithm for attack localization in networks. The algorithm is distributed and requires only local information. The algorithm can localize attacks for a variety of network applications. This algorithm is particularly well suited to the requirements of All-Optical Networks because it provides fast, reliable response to attacks. In particular we apply it to two common forms of rapid optical network restoration: automatic protection switching and loopback.

1. Introduction

All-optical networks (AONs) are emerging as the networks of choice for ultrafast (over 1 terabit per second [15]) communications and are being demonstrated in several testbeds [19, 50, 16, 4, 5, 1, 10]. While the architectures and implementations of AON testbeds vary, they all share certain common hardware building blocks and functional characteristics. As the name indicates, AONs do not contain electronic processing components, thereby avoiding electronic bottlenecks. Signals which flow through an AON undergo only all-optical switching (which affords network functionality) and all-optical amplification (which counteracts attenuation of the optical signals through the network).

The devices which perform switching and amplification are rapidly maturing but have certain drawbacks. In particular, they exhibit "crosstalk" characteristics, so that a nefarious user on one channel can, by exploiting the physical properties of switches and amplifiers, affect other channels whose routes share devices with the nefarious user's channel. Since the nefarious user's signal flows unchecked through the AON, an attacker may use a legitimate means of accessing the network to effect a service disruption attack, causing a quality of service degradation or outright service denial. The operation of AON components have important security ramifications [30].

In this paper we present an algorithm that finds the origin of the attacking signal. By localizing the attack the network maintains quality of service whereas an algorithm that localizes component failures would result in service degradation or denial (for example see sections 1.3 and 4). While the algorithm we present is particularly necessary for localization of propagating attacks, it will also localize component failures which we can view as non-propagating attacks.

In the remainder of this section, we motivate the need to provide algorithms for attack localization in AONs, present a model of our problem and give an overview of previous research on attack localization in networks, in particular research that applies to AONs. In section 2, we present our algorithm. In section 3, we discuss applications of the algorithm to attack localization. In section 4, we show how our algorithm can be used for service restoration after an attack for two important types of preplanned recovery schemes: automatic path protection switching and loopback. Finally, in section 5, we present our conclusions and directions for further research.

1.1. AON Vulnerability to attack

One of the main security issues for optical networks is that service disruption attacks can spread through a network. We need to be able to differentiate between a failure and an attack and we must be able to locate the source of an attack. An overview of culnerabilities is given in [30]. For instance, at an amplifier, a user with a particularly strong signal can rob other users' signals of power. Such a nefarious user can disrupt several users who share amplifiers with him. We term this type of attack a gain competition attack. In another example of malicious use of the network, a switching device can be used to insert a portion of one channel's signal onto another channel's signal. The two attacks may be combined as shown on figure 1. Channel 1 uses crosstalk at the switch to rob channel 3 of its power at the amplifier. Note that channel 1 can be used to affect channel 3, even though channels 1 and 3 are routed through distinct components.

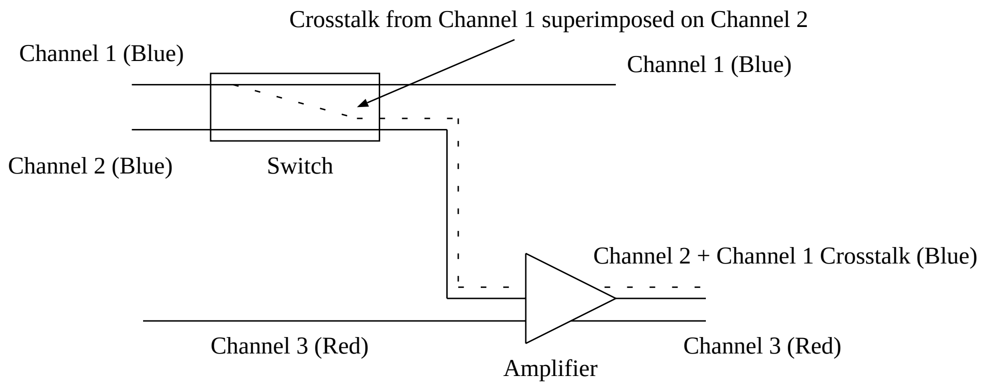


Figure 1. Combined switch and amplifier attack.

In circuit-based networks with very high rates, ultrafast restoration is preplanned and based upon local information, in order to avoid the delays associated with software processing. In order to allow the network to recover from attacks, we must be able to identify attacks carried out by network traffic and to localize these attacks. If we were to apply identification and localization methods which are designed for naturally occurring failures to the case of service disruption attacks in AONs, an attack at a single point might lead to widespread failures.

Let us illustrate why it is important to be able to identify an attack caused by the traffic itself from a failure which occurs because of natural fatigue of components or physical sabotage of the network. For instance, let us consider the case where recovery of a node failure is performed by rerouting traffic away from that node. An example of such a node failure recovery scheme is the SONET/SDH bidirectional self-healing ring. If the traffic itself is the cause of the failure, as is the case in the amplifier and switch attacks discussed above, then failures will be caused throughout the network without any restoration. Consider an attack on node i, which carries channels 1, 2 and 3, from channel 1. If the network management deals with all failures as though they were benign failures, then it assumes that node i failed of its own accord and reroutes the three channels to some other node, say j. After that rerouting, j will appear as having failed because channel 2 will attack j. The network may then reroute all three channels to node k, and so on. Therefore, it is important for node i under attack to be able to identify an attack coming from its traffic stream and to differentiate it from a physical hardware failure which is not due to the traffic streams traversing i.

It is insufficient to differentiate an attack carried out by the network traffic from a physical failure. We must be able to identify the source of the attack, as the example in figure 2 illustrates. In this example, Channel 1 is attacking the network by sending an excessively powerful signal. Let us suppose that each node guards against jamming attacks by disconnecting a channel which is identified as being too powerful. As shown in figure 2, channel 1 and channel 2 both share the same node i, in this case a switch, and are both carried on a blue wavelength. Crosstalk from channel 1 is superimposed upon channel 2 at node i. Channel 2, in turn, may become too powerful and disrupt channel 3, also on blue, at node j, which is also a switch. Nodes i and j may both correctly identify the failure as a crosstalk jamming attack. Node i will correctly identify the offending channel as channel 1 but node j will identify the offending channel as channel 2. If the network has no means of localizing the source of the attack, then node i will disconnect channel 1 and node j will disconnect channel 2. Channel 2 will therefore have been erroneously disconnected.

1.2. Example of service denial due to attack

We now illustrate how an attack of the type discussed in the above section could lead to service denial. The ability to use attacks to deny service stems from the fact that attacks spread, causing malfunctions at several locations, whereas failures generally do not disrupt the operation of several devices. Thus, while a single network element failure may cause several network elements to have corrupted inputs and outputs, the failure will not generally cause other network elements to be defective in their operation. Because of transparency we do not have an absolute metric to determine whether an input is faulty or not. Instead we look at the operation of a node, i.e., the relation between the input and the output. An attack will lead to incorrect operation of the node. An attack, as illustrated in the previous section, can cause network elements not only to have cor-

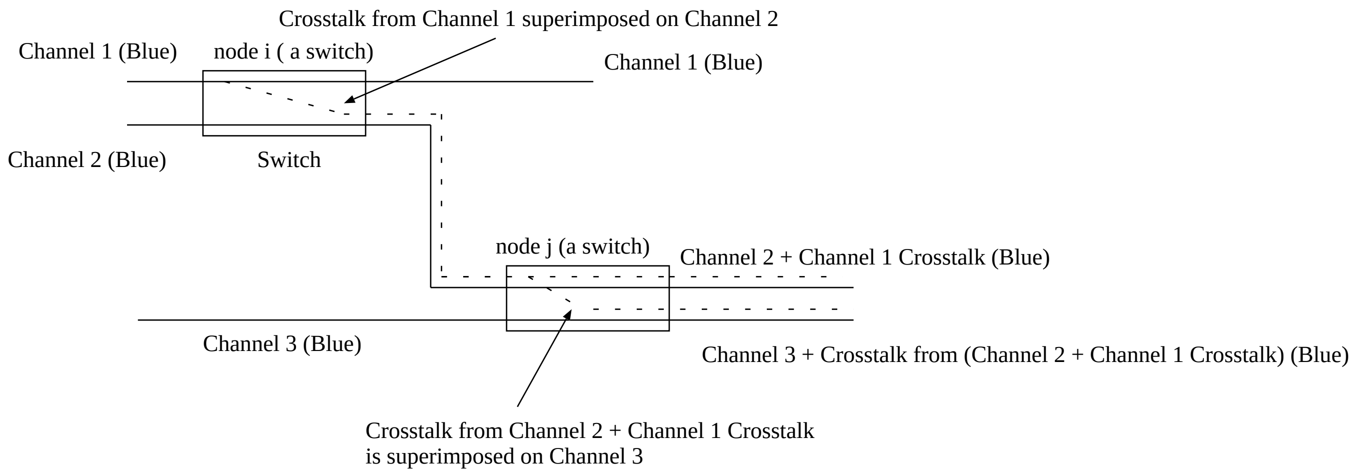

Crosstalk from Channel 1 superimposed on Channel 2
Channel 1 (Blue)
node i (a switch)
Channel 1 (Blue)
Channel 2 (Blue)
Switch
node j (a switch)
Channel 2 + Channel 1 Crosstalk (Blue)
Channel 3 (Blue)
Channel 3 + Crosstalk from (Channel 2 + Channel 1 Crosstalk) (Blue)
Crosstalk from Channel 2 + Channel 1 Crosstalk
is superimposed on Channel 3

Figure 2. Attack through two switches.

rupted inputs and outputs, but the nature of those corrrupted inputs can lead to improper operation of the network elements. Hence, if alarms are raised at individual network elements by improper operation of the network element, a failure will lead to a single alarm. An attack, on the other hand, may lead to several alarms downstream (in the flow of communications) of the first point which is attacked.

If a scheme is prepared to recover from failures but encounters instead an attack, it may malfunction and cause failures. A common recovery scheme is based on rings. We consider the two types of network restoration used for SONET/SDH, the most common standards in use for high-speed optical communications [55]. Note that SONET/SDH are not all-optical standards, but the rates they support make their need for rapid service restoration commensurate with that of AONs. Speed of recovery with opto-mechanical switches is in the tens of milliseconds and nanoseconds for acousto-optical switches. SONET/SDH allow for network restoration after failure in two ways:

- By having two streams traverse physically node (or link) disjoint paths between a source and a destination, as in the SONET UPSR (unidirectional path switched ring) approach. In case of failure of a node (link) along one stream, the receiving node listens to the redundant, backup, stream. We term this approach an automatic protection switching (APS) approach.
- By having a single stream being rerouted onto a backup channel in case of a failure, as in the SONET BLSR (bidirectional line switched ring) approach. We term this approach the loopback approach.

Figure 3 shows the two approaches.

The fact that, for any node (edge) redundant graph, there exists a pair of node (edge)-disjoint paths, that can be used for APS, between any two nodes is a consequence of Menger's theorem [44, 31]. There have been a variety of proposed path rerouting schemes based on Menger's theorem, e.g. SNCP and different variants of it [2, 39, 45, 59, 53, 27]. Automatic protection switching over arbitrary redundant networks need not restrict itself to two paths between every pair of nodes, but can instead be performed with trees, which are more bandwidth efficient for multicast traffic [29, 11, 52, 18]. For loopback protection, most of the schemes have relied on interconnection of rings or on finding ring covers in networks [57, 51, 56, 42, 43, 40, 41, 12, 58, 47, 54] and [55, pp. 315–325]. Loopback can also be performed on arbitrary redundant networks [28].

Let us show how a single attack may lead to a failure in the case of loopback recovery. Figure 4 illustrates our discussion. Let us denote by j the attack source. Thus, node j is attacked, for instance by a nefarious user who uses j as a point of entry into the network for insertion of a spurious jamming signal. The jamming signal causes the nodes adjacent to j to infer that j has failed, or is "down". The same jamming signal, upon travelling to k, will cause the nodes adjacent to k to infer that k has failed. If both j and k are considered as individual failures by the network management, then loopbak will be performed to bypass both j and k in a ring. Thus, all traffic which passed through both j and k will be disrupted, as shown in figure 4. If, instead, j is correctly identified as the source of the attack, then loopback effected to bypass j will lead to correct operation of the network, with only the inevitable loss of traffic which

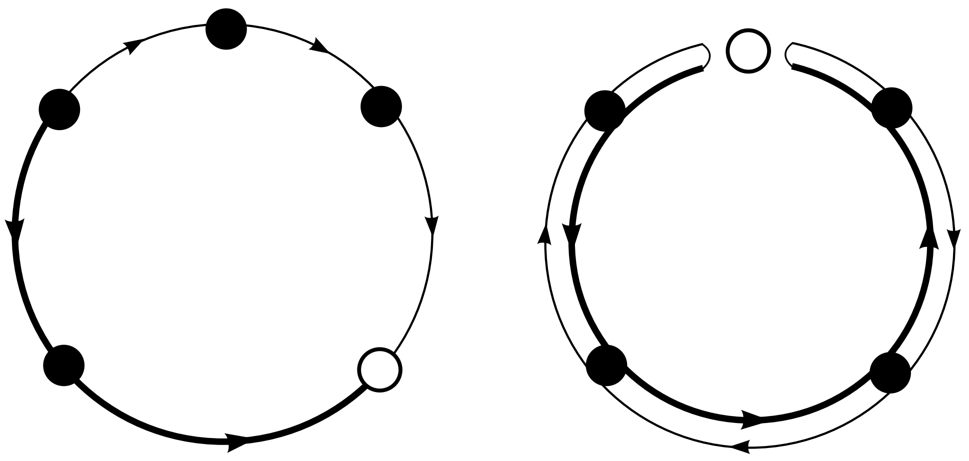


Figure 3. Automatic protection switching and loopback protection. Black nodes show operational nodes, white nodes show failed nodes, thin lines show primary routes, and thick lines show back-up or restoration routes.

had j as its destination or origination.

In section 4 we show how to apply the attack localization algorithm to loopback recovery to avoid such unnecessary service denial.

1.3. Problem Statement

We have seen the need to identify and localize attacks. We do not examine here the means available for detection and identification of attacks. Instead, we consider that attacks can be detected and identified with satisfactory false positive and false negative probabilities. Recall that an attack is a malfunction which affects the input to output relation at the node. We seek to create an algorithm which can rapidly identify the source of an attack. For instance, in the example depicted in figure 2, if node j knows that node i also had a crosstalk jamming attack on blue, it allows i to disconnect the attacker. Once node i disconnects channel 1, channel 2 ceases to appear as an offending channel at node j. If node j does not have information from node i indicating that channel 1 is an attacker at i then node j infers that channel 2 is the attacker at node j. Node j then disconnects channel 2. Note that node j sees no difference between the cases where channel 1 is the attacker at i and where channel 2 is the attacker at j. In both cases, channel 2 appears as the attacker at j. By using knowledge from the operation of node i upstream of j, j can deduce whether the attack originated with channel 1 or channel 2.

We consider a network composed of nodes and links. Each node handles a certain number of channels. Channels may terminate or originate at certain nodes. Each channel has a specific direction and we may therefore speak of nodes being upstream or downstream of one another for a certain channel. Each node is able to detect and identify attacks being levied against it, receive and process messages arriving to it and generate and transmit messages to nodes which are upstream or downstream of it on certain channels. Note that a node in our model may not correspond to a network component. For instance, we can model a switch as several nodes, one for each switching plane and component amplifier. Conversely, a cascade of in-line amplifiers may be modeled as a single node because they have a single input and a single output.

We must explicitly take into account the time taken by the different processes involved in the identification and localization of attacks. The identification of an attack requires time for detection of the input and output signals and processing of the results of that detection. There is also delay involved in generating messages to upstream and/or downstream nodes. We denote all the time required by all of the above processes executed in sequence as the processing time τ_i^{meas} at node i. Messages from node i to node j take time T_{ij} to transmit. Message transmission follows the transmission of the data itself, and does not usually add to the overall time of localizing the attack. Lastly, there are delays due to the time for capturing messages from upstream and/or downstream nodes, the time to process these messages together with local information and the time to

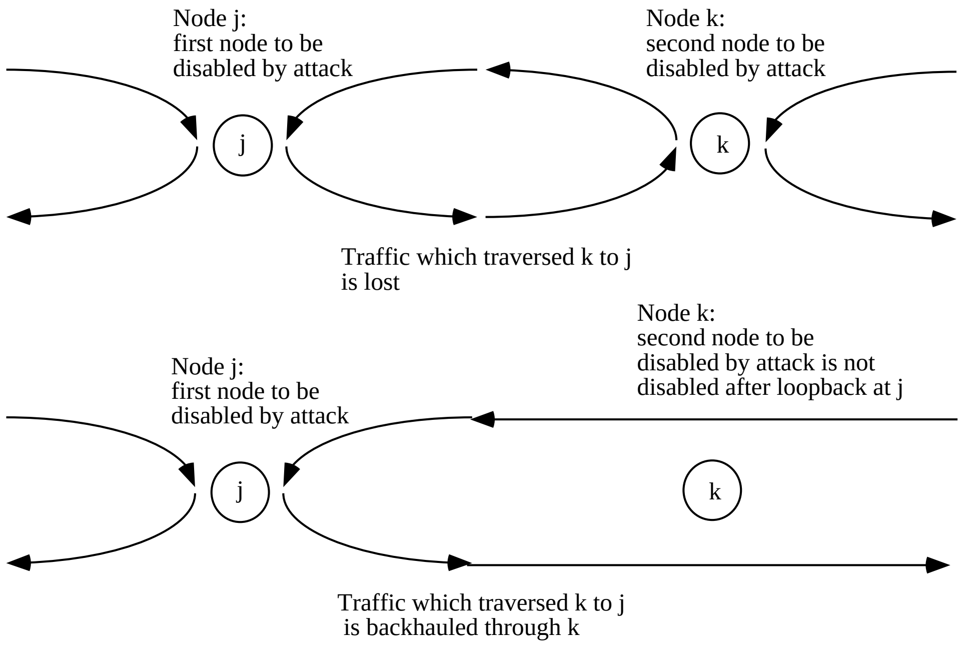


Figure 4. The top figure shows loopback recovery when both nodes j and k detect an attack and both nodes are believed to be faulty. The bottom figure shows the more appropriate recover scheme for an attack at j which propagates to k. Here only node j is rerouted and traffic can reach the now unaffected node k.

generate new messages. We denote the time required by this last set of events as τ_i^{proc}.

Our goals are:

Localization Localization of the source of an attack to enable automatic recovery.

Speed Very fast operation (implying near constant operational complexity).

Scalability The delay must not increase with the size and span of the network.

Robustness Valid operation under any attack scenario including sporadic attacks.

We now consider known methods of fault localization and determine their applicability to our problem.

1.4. Related Work

There has been much work in the area of fault localization in current data networks, which are often packet data networks, but not all of those results are applicable to optical networks. We do not seek to provide an exhaustive overview of the subject, but simply to compare the main thrust of this work to the existing literature. We give a brief overview of three different sets of fault diagnosis frameworks: fault diagnosis for computing networks, probabilistic fault diagnosis by alarm correlation and fault diagnosis methods specific to AONs.

The first topic covers the case where we have units which communicate with subsets of other units for testing. Each unit is permanently either faulty or operational. The test on a unit to determine whether it is faulty or operational is reliable only for operational units. The problem of diagnosing such systems was introduced in [37]. Necessary and sufficient conditions for the testing structure for establishing each unit as faulty or operational as long as the total number of faulty elements is under some bound were found in [14]. Polynomial-time algorithms for identifying faults in diagnosable systems have been found [7, 8, 13]. Instead of being able to determine exactly the faulty units, another approach has been to determine the most likely fault set [26, 6]. All of the above approaches have several drawbacks with respect to the goals we seek to achieve:

- They require each unit to be fixed as either faulty or operational. Hence, sporadic attacks which may only temporarily disable a unit cannot be handled by the above approaches. Thus robustness is not achieved.

- They require tests to be carefully designed and sequentially applied [48, 49, 22, 32]. The number of tests required, moreover, rises with the possible number of faults [23, 14]. Thus the scalability goal is not achieved.
- The tests do not establish any type of causality among failures, hence cannot establish the source of an attack by observing other attacks. Thus the first goal of localization is not achieved.
- Fault diagnosis by many successive test experiments may not be rapid enough to perform automatic recovery, thus violating our speed goal.

Another approach is related to Baysian analysis of alarms in networks. Alarms from different network nodes are collected centrally and analyzed to determine the most probable failure scenario. Unlike the schemes discusses in the previous paragraph, this type of analysis can be used to discover the source(s) of attacks thus complying with the first goal. Moreover, it can analyze a wide range of time-varying attacks in accordance with the robustness goal. A good treatment of alarm correlation for diagnostic of failures and of related work can be found in [21]. A general treatment of the Bayesian analysis involved in diagnostic problem solving is given in [34, 35]. All of the above results assume some degree of centralized processing of alarms, usually at the network and subnetwork level. The time complexity of the software processing grows in some fashion with the size of the network. Moreover, there are delays involved with propagation of the messages to the processing locations. Hence, the solutions may not scale well as the data rates increases or the size of the network grows violating the scalability and speed goals. If either the data rate or the span of network increase, there is a growth in the latency of the network, i.e. the number of bits in flight in the network. The combined increase in processing delay and in latency implies that many bits may be beyond the reach of corrective measures by the time attacks are detected. Therefore, an increase in network span and data rate would lead to an exacerbation of the problem of insufficiently rapid detection.

For AONs, fault diagnosis [20, 9, 38, 24] and related network management issues [25, 3] have been considered. Some of the management issues for other high-speed electro-optic networks are also applicable [33, 36]. The problem of spreading of fault alarms, which exists for several types of communication networks, is exacerbated in AONs by the fact that signals flow through AONs without being processed [25]. If we are only concerned about fiber failure, then only the nodes adjacent to the failed fiber need to find out about the failure and a node need only switch from one fiber to another [17]. For failures which occur in a chain of in-line repeaters which do not have the capability to switch from one fiber to another, an approach is given in [46]. When a failure occurs, the alarm due to the failure is generated by the in-line repeater immediately after the link failure. The failure alarm then travels down to a node which can perform failure diagnostic. The failure alarms generated downstream of the first failure are masked by using upstream precedence. Failure localization can then be done by having the node capable of diagnostic send messages over a supervisory channel towards the source of the failure until the failure is localized and an alarm is generated at the first repeater after a failure. We use the idea of precedence of upstream failure attacks but we do not require diagnostic to be perfomed by remote nodes and to have two-way communications between nodes. In the following sections, we discuss our scheme and show the benefits of not requiring two-way communications.

2. Attack Identification and Localization Algorithm

The algorithm we develop for attack localization is distributed, and uses local communication between nodes up- and down-stream. Each node in the network determines if it detects an attack. It then processes messages from neighboring nodes to determine if the attack was passed to it or if it is the first node to sustain an attack on a certain channel. We denote the first node affected by an attack as the *source of the attack*, even though the attack may have been launched elsewhere. The global success of localizing the attack depends upon correct message passing and processing at the local nodes.

The main thrust of our algorithm is the recognition that, in order for a node to determine whether or not it is the source of an attack, it need only know whether a node upstream of it also had the same type of attack. Suppose that node i is upstream of j on a certain channel which is identified as being an attacking channel and that both i and j identify the attacking channel. Suppose that both i and j have processing times τ^{meas} and τ^{proc}. If i transmits to j its finding that the channel is nefarious, then the interval between the time when the attack hits j and j hears from i that the attack also hit i is at most τ^{meas}. Indeed, the attack and the message concerning the attack travel together. Moreover, the detection and identification of the attack commences at j as soon at the attack hits. Hence, the elapsed time until j identifies the attack and determines whether i also saw that attack is $\tau^{meas} + \tau^{proc}$. Note that this is independent of the delay in the communications between i and j because the attack and the message concerning the attack travel together, separated by a fixed delay. If the attack hits several nodes, each node only waits time $\tau^{meas} + \tau^{proc}$ to determine whether or not it is the first node to detect that attack, i.e. whether it is the source of the attack.

To illustrate the main points of the algorithm we first consider a simple attack localization problem. In this network nodes can either have a status of 1 (O.K.) or 0 (alarm). Nodes monitor messages from upstream. Let the message be the status of the node. When an attack occurs in this network, the goal of this algorithm is that the node under attack respond with an alarm and all other nodes respond with O.K.

Each node in the network repeats the following algorithm continually and responds accordingly.

Algorithm 1 Basic Node Protocol

```
Compute the status, $s$ of the node at this time, $t$
Transmit message $s$ to all the adjacent nodes downstream
If $s \neq 1$ then
    Let $InMessages$ = all messages arriving in the time
        interval $[t, t + \tau^{meas}]$
    If any message in $InMessages \neq 1$ then
        then set $s = 1$
        else set $s = 0$
```

We can immediately see that no node will generate an alarm until at least one attack is detected. When an attack occurs only the first node experiencing the attack will respond with an alarm. All nodes downstream from this node receive messages which indicate that a node upstream experienced an attack. Thus, nodes downstream from the attack will respond with O.K. This network response achieves our stated attack localization goal. In section 3.1 we revisit this simple network problem with a more rigorous algorithm statement. We then show that attacks do not propagate in this network.

We now turn to the task of rigorously defining a general form of the network localization algorithm.

2.1. Definitions

This section provides the definitions and notation that we will use in the remainder of this paper.

Network Definition

A network has nodes $1, 2, ..., n$. Each node has inputs I_{ij} and outputs O_{ij}. We say the network has directed connection (i, j) when the is a connectin from node i to node j by a link. We will refer to the undirected connection between nodes i and j as $[i, j]$. We assume that the network is acyclic.

Time Delays

Recall that we define the time delays for the processing and transmission as follows:

τ_i^{meas} = measurement time for node i including time to format and send messages.

τ_i^{proc} = processing time for nodes i including time to format and send messages.

T_{ij} = time to transmit on arc (i, j). Assume wlog that $T_{ij} = T_{ji}$.

In many of the examples we discuss, the time delays at all nodes are identical. We then refer to the measurement and processing time as τ^{meas} and τ^{proc} without subscripts.

Messages and Faults

We assume that each node can detect an attack or fault within acceptable error levels. Let the fault types be in the set F. One of the fault types in F is always the no fault case.

The status of a node at time t is $S_t(i) \in F$. We will use $S(i)$ to indicate the current status of node i.

Let us consider connection (i, j). A *message* from node i to node j at time t is denoted $M_t(\overrightarrow{i,j})$. Messages can be sent up- or down-stream in the network. The upstream message from node j to node i at time t is denoted $M_t(\overleftarrow{i,j})$. For particular network applications the information encoded in messages varies. However, messages should remain small for fast transmission and processing such as the messages passed by algorithm 1. For example, a node can transmit its status up- and down-stream via the messages $M_t(\overrightarrow{i,j}) = S_t(i)$ and $M_t(\overleftarrow{i,j}) = S_t(j)$. Message $M_t(\overrightarrow{i,j})$ arrives at node j at time $t + T_{ij}$ and likewise message $M_t(\overleftarrow{i,j})$ arrives at node i at time $t + T_{ij}$. Again, we will write $M(\overrightarrow{i,j})$ and $M(\overleftarrow{i,j})$ to indicate the current message from node i to j and from node j to node i, respectively.

Response Function

The crux of the algorithm lies by the response function, R. This function processes incoming messages and local status information to determine the response of the node. We will discuss this function is the context of the algorithm in the following section.

2.2. The General Algorithm for Attack Identification and Localization

The general algorithm, like the simple example of attack localization we discussed earlier, is a distributed algorithm that achieves its goal through local processing and message passing. The goal of the algorithm can vary for different network examples. For example, the goal may be to raise an alarm as in algorithm 1. A more complex goal may be to reroute the node immediately before and after the attacked node in the network. The following statement of the algorithm is general enough to be suitable for a wide range of network goals. The algorithm achieves this generality by leaving many of the details of the algorithm (such as the set of faults, the format of the messages and the node response to input messages) unspecified in the general algorithm statement. These specifics must be defined for the particular network application.

Algorithm 2 General Node Protocol

```
Compute $S_t(i) = f_i$
For all $(i,j)$ transmit messages $R_i(S(i), \emptyset)$
Let $InMessages$ = all messages arriving in the
    interval $[t - T_i^{wait1}, t + T_i^{wait2}]$
Transmit $R_i(S(i), InMessages)$
```

This algorithm ascertains the fault type and transmits it to adjacent nodes in the network. It then monitors incoming messages for a specified (bounded) time interval and responds to these messages. The response of the network is unspecified in the above statement. For generality, we replaced the processing and response with the response function, R, which we will discuss shortly. To achieve a particular network application, the following must be set:

- The fault set, F.
- The waiting time interval for messages, i.e. T^{wait1} and T^{wait2}.
- The format of messages.
- The response function, R.
- The mode of message passing. The node can remove messages it receives from the message stream or pass all messages in the message stream.

The Response Function R

In the general algorithm the response function R is responsible for achieving the data transmission and security goals of the network.

R is a function: $Status \times MessageList \longrightarrow MessageList$. Function R should be very fast to compute in order to satisfy our speed goal. Ideally the function should consist of a few compares and table lookups. Thus the delay in identifying faults and attacks is short and the network provides minimal data loss.

Messages can move up- or down-stream in the network. The response function receives all the messages as input. It processes these messages to generate the messages for transmission from the node. The response function generates messages which the node transmits up- and downstream. As we will see in sections 3 and 4, the response function can be defined to handle a variety of network recovery applications.

In addition, function R may have a side effect response, such as raising an alarm or re-routing traffic at a node.

Each node, i, in the network can have a different response function, R_i. The use of different response functions, with varying processing times, may, however, result in race conditions in the network. In general we can avoid timing problems due to different response functions by forcing all response functions in a network to operate in the same amount of time. We set the processing time to be the maximum time required by any of the response functions. We then add a wait time to each response function such that its final processing time is equal to the maximum time.

Note that the response function may return no message, or the empty set, in which case no messages are transmitted.

3. Example Applications

3.1. Basic Attack Identification and Localization

We now re-consider the problem of basic attack localization of algorithm 1 and implement this algorithm in the framework described in the previous section. Recall that, for this problem, the nodes have two fault types: no fault and fault (i.e. $F = \{1, 0\}$), the status of a node is $S(i) \in F$, and messages from any node encode the status of the node. The goal for node i is to determine whether it is the source of the attack or if the attack is being carried by the data from a source upstream.

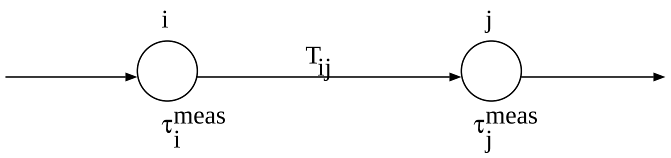
We set, in the general algorithm, the waiting times $T^{wait1} = 0$ and $T^{wait2} = max_i(\tau_i^{meas})$ and the message passing parameter to remove all messages received. The response function R is as follows:

Algorithm 3 R for basic attack localization
Input: s, $InMessages$

```
If $InMessages = \emptyset$ then return $s$
else if $s \neq 1$ then
    If $\exists M(\overrightarrow{j,i}) \in InMessages$ s.t. $M(\overrightarrow{j,i}) \neq 0$
        then return $1$
        else return $0$
```

We can immediately see that the above response function results in the basic attack localization algorithm (algorithm 1). We will now show that this algorithm achieves our stated goal.

In localizing the attack, we look at the dynamics between two nodes and the connection between them. Each node monitors every connection into it. For the simplest case, we examine a connection between nodes i and j, with the data flowing from i to j.



Let the time at which the data leaves node i be 0. The message from node i to node j about i's failure is sent at time τ_i^{meas}. Node j receives the data at time T_{ij}, and completes measurement and sends its status at time $T_{ij}+\tau_j^{meas}$. At this time node j has detected an attack or it has detected no attack. Node j receives the message from node i at time $T_{ij}+\tau_i^{meas}$. Thus, node j can begin to process the status message from i at time $T_{ij}+max(\tau_j^{meas},\tau_i^{meas})$. At this time node j knows whether or not node i detected an attack, and node j has enough information to determine the whether or not it is the source of the attack. Processing at j falls into one of four cases.
Case 1: If j has a detected no attack, then j concludes that it is not the source of an attack.
Case 2: If j has detected an attack and i has detected no attack, then j concludes that it is the source of the attack.
Case 3: If j has detected an attack and i has detected a attack, then j concludes that it is not the source of the attack.
Case 4: If j has detected an attack and has not heard from i at time $t = max(\tau_i^{meas},\tau_j^{meas})+T_{ij}$, then j concludes that it is the source of the attack.

Node j completes processing at $T_{ij}+max(\tau_j^{meas},\tau_i^{meas})+\tau_j^{proc}$

We can show, by an exhaustive enumeration of the possible timing constraints involving τ_i^{meas}, τ_j^{meas}, τ_j^{proc}, T_{ij} and the length of the attack L, that node j is never in the wrong state, i.e. it concludes at time $t+max(\tau_j^{meas},\tau_i^{meas})+\tau_j^{proc}$ that it is the source of an attack if and only if it is the source of an attack at time t.

3.2. Localization of Attacks that Affect Selected Nodes Downstream

In this section we consider a specific attack scenario due to crosstalk. Usually, in the case of an attack which is carried by the signal, we assume that all the nodes through which the signal is transmitted will be affected by the attack, i.e. they will suffer a fault. The basic attack localization algorithm described in section 3.1 can localize the source of such attacks.

The problem we consider now involves an attack which is carried by the signal, but may not be detectable in some nodes. As the signal traverses down the network it attacks some nodes then reaches a node which it does attack and continues on to attack downstream nodes. For example, consider an attack of channel 1 at node i, a switch, in the network nodes of figure 5. Owing to crosstalk at i, the output of channel 2 at node i is affected by the attack. The signal in channel 2 then transmits to node j, which is an amplifier. Since this signal is the only input to node j, gain competition is not possible so this node does not detect an attack. At node k, however, channel 3 is once again affected by crosstalk from the attack, thus an alarm is generated. The attack does propagate. It is detected in nodes i and k, but it is not detected at intermediate node j.

We want to apply the attack localization algorithm to this problem. To isolate the salient issue we will consider the simplest framework within which this problem can occur. We therefore deviate as little as possible from the framework in section 3.1. Nodes have two fault types, no fault and fault, and the message simply contains a status: fault or no fault. The goal of the algorithm is also unchanged, node i must determine whether it is the source of the attack or if the attack is being carried by the data from a source upstream.

The difference between this problem and the basic attack localization problem is that each node must know of the status at all the nodes upstream from it in the network, whereas in the basic attack localization problem we assumed that when an attack propagates every node in the network detects a fault so the status from the single preceding node contains sufficient information from which to draw conclusions. Instead of generating messages at each node, the data is followed from its inception by a status message which lags the data by a known delay. The status message is posted by the node at which the communication starts. Once an attack is detected the status message is disabled. The lack of a status message indicates to all the nodes downstream that the source of the attack is upstream of them. Note that such a status message is akin to a pilot tone associated with the data stream. The lack of a pilot tone indicates that an attack or fault has occurred.

We can now define the response function, R, from the general algorithm.

Algorithm 4 **R for selective attack localization**
Input: s, $InMessages$

```
If $InMessages = \emptyset$ then return $\emptyset$
else if $s \neq 1$ then
    then disable status message.
        return $\emptyset$
return $\emptyset$
```

Note that the nodes in the network never generate message. They can, however, disable the status message when they detect an alarm. When the status message is disabled then any node downstream can conclude that it is not the origin of the attack.

We set, in the general algorithm, the waiting times $T^{wait1}=0$ and $T^{wait2}=max_i(\tau_i^{meas})$ and the message passing mode is to transmit all messages.

Suppose a node i is attacked at time t. It will turn off the status message at time $t+T^{wait2}+\tau^{proc}$. The next node, j, receives the data stream at time $t+T_{ij}$ and waits until time $t+T_{ij}+T^{wait2}$. We immediately see that a race condition may arise if $\tau^{proc} > T_{ij}$. For an all-optical network, switching off a channel can be done in the order of

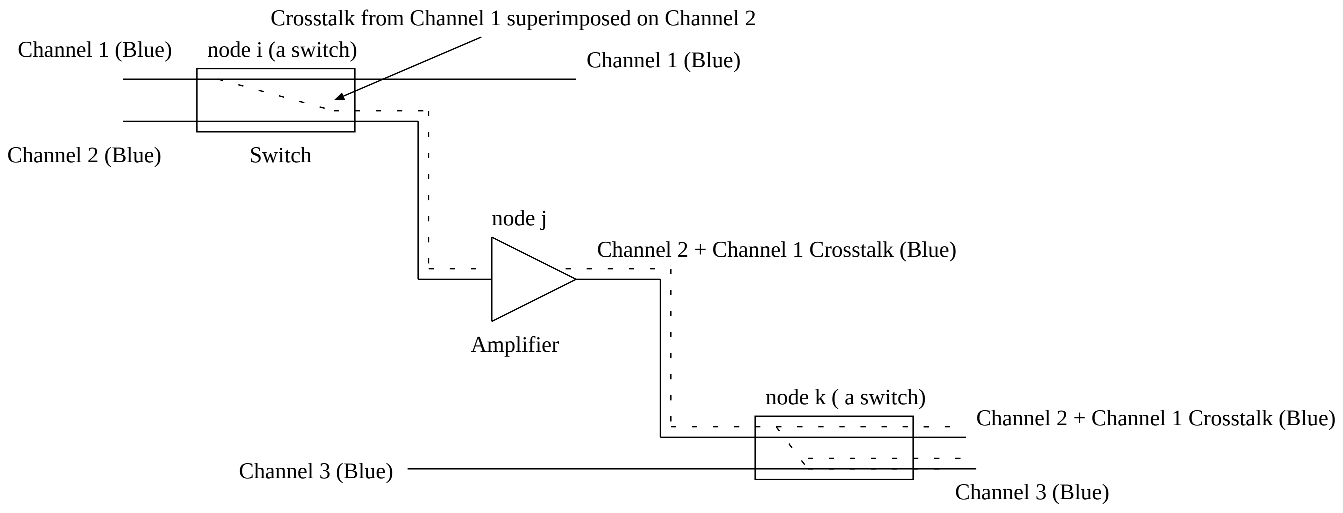


Figure 5. A propagating attack which does not disrupt all nodes.

nanoseconds with an acousto-optical switch. The delay between nodes in the network would typically be larger, so we do not believe this condition will be problematic in practice. Moreover, the network can be designed to ensure this condition is met by introducing delay at the nodes. Such a delay is easily obtained by circulating the data stream through a length of fiber.

3.3. Multiple fault types

Response to multiple fault types can be handled efficiently with a lookup table. In this case the response function R would have a pre-stored table L. Given the current status, s_i, and the status of the previous node, s_j, the look-up table provides the appropriate response for this node, r_i (i.e., $L : Status \times Status \longrightarrow Response$.) For some applications it is useful to have different lookup tables for the next node the network, L_n, and the previous node in the network, L_p. Furthermore, the look-up tables can be extended to the domain of $Status \times Response$ which gives greater flexibility.

3.4. Repressing Alarms for Corrected Signals

Consider a node which detects signal degradation. The signal may be amplified sufficiently by the next node downstream to remain valid when it reaches the destination. We would not want to stop transmitting the signal or to re-route the node that detected this problem. Instead, network operation should continue as usual and an alert, but not an alarm, should be generated. We have three possible response values: $O.K.$, $alarm$ and $alert$. An $alarm$ is generated by the source of an attack which is not corrected, whereas an $alert$ is created by the source of a corrected attack.

Our attack localization algorithm can achieve this behavior using upstream messages. Each node must send status messages upstream as well as downstream. Upon detecting an attack in a node downstream, messages are checked to determine if this node is the source of the attack. Upstream messages are checked to determine if the attack persists in the next node downstream. When a node detects an attack is first generates an alarm. If it later finds that the problem was corrected downstream it downgrades its alarm to an alert.

The response function for this network is shown in figure 6.

Upstream messages follow the data stream by a significantly longer time than do downstream messages. An upstream message requires time for the data to traverse a link (i, j) to the next node, j. The status of node j must be measured, and the message from node j to node i must traverse the link (i, j). Therefore the waiting time, T^{wait2} in the attack localization algorithm is longer when upstream messages are monitored. In particular, for this scenario we need $T^{wait2} = 2 * max(T_{ij}) + max_i(\tau_i^{proc})$.

4. Application to Network Restoration

In this section, we present how our algorithm can be used for network restoration. We consider the two types of network restoration used for SONET/SDH: automatic protection switching and loopback (see discussion in section 1.2). For each network restoration scheme we describe how our algorithm can be used to perform recovery and provide the algorithm that achieves the attack localization.

Algorithm 5 R **for repressing alerts**
Input: s, $InMessages$

```
If $InMessages = \emptyset$ then
    if $s = 1$
        then return $O.K.$
        else return $alarm$
else if $s \neq 1$ then
    Let $DownstreamMessages = \{m \in InMessages$ s.t. $m = M(\overrightarrow{j,i})\}$
    Let $UpstreamMessages = \{m \in InMessages$ s.t. $m = M(\overleftarrow{i,j})\}$
    If $\exists M(\overrightarrow{j,i}) \in DownstreamMessages$ s.t. $M(\overrightarrow{j,i}) \neq O.K.$
        then return $O.K.$
        else If $\exists M(\overleftarrow{i,j}) \in UpstreamMessages$ s.t. $M(\overrightarrow{j,i}) \neq O.K.$
            then return $alarm$
            else return $alert$
```

Figure 6. Response function,R, for repressing alerts

4.1. Application to automatic protection switching.

APS allows the network to receive data on the backup stream on the event of a faulty node. In the case of an attack, service would be maintained if the attack is detected. However the location of the attack is unknown and restoring normal network operation may require a great deal of time.

The attack localization algorithm (algorithm 3) from section 3 gives the network the required information to switch streams upon an attack or a fault. Furthermore, the attacked node is identified so that the attack can be dealt with quickly.

The basic fault localization algorithm can be used to find out whether or not an attack took place along the primary path. Figure 7 shows the primary and the backup paths in a network. If an attack took place along the primary path, there will be a message indicating the presence of such an attack and lagging the attack by τ^{meas} traveling alongside the primary path. The end node will therefore know that there was an attack upstream and that the destination node, d, was not the source of the attack. The response of the destination node, d, will be to listen to the backup stream.

This network requires two response functions: one for destination nodes, R_d, and one for all other nodes, R_n. We can set R_n to R in algorithm 3. The destination node response function is

Algorithm 6 R_d **for destination nodes in APS**
Input: s, $InMessages$

```
If $InMessages = \emptyset$ then return $s$
else if $s \neq 1$ then
    Receive data on backup stream.
```

Since the attack localization algorithm relies on messages arriving at nodes at specific times, we are concerned that two different response functions may not obey these timing conditions. Since all nodes in the network except the destination nodes use R_n, the timing up to the destination node will not result in race condition. Since no node is waiting for messages from the destination nodes, any differences in time will not affect nodes in the network.

Switching routes can entail certain routing problems. We can avoid such problems by delaying transmission on the backup path. We now compute the necessary delay on the backup path. Let us denote by τ^{switch} the time it takes for d to switch from the primary path to the backup path after an alarm on the primary path has been diagnosed by d. Let $\Delta\tau$ be the difference in transmission delay between the source, s, and the destination, d, between primary stream and backup stream. We assume that the transmission delay is shorter on the primary path and longer on the backup path. Regardless of where the failure happened on the primary path, we may see that no data will be lost in the process of detecting the problem and switching to the backup stream as long as the data on the backup stream is transmitted with a delay of at least

$$max_{\text{all nodes in the primary path}}(\tau_i^{meas}) + \tau^{switch} - \Delta\tau.$$

If all nodes have the same τ_i^{meas}, then, no matter where the failure occurs in the primary path, there is always the same delay between the primary data stream and the backup data stream after APS. Therefore, we do not need the destination node, d, to adapt its response to the location of the failure. Independence from the location of the failure is

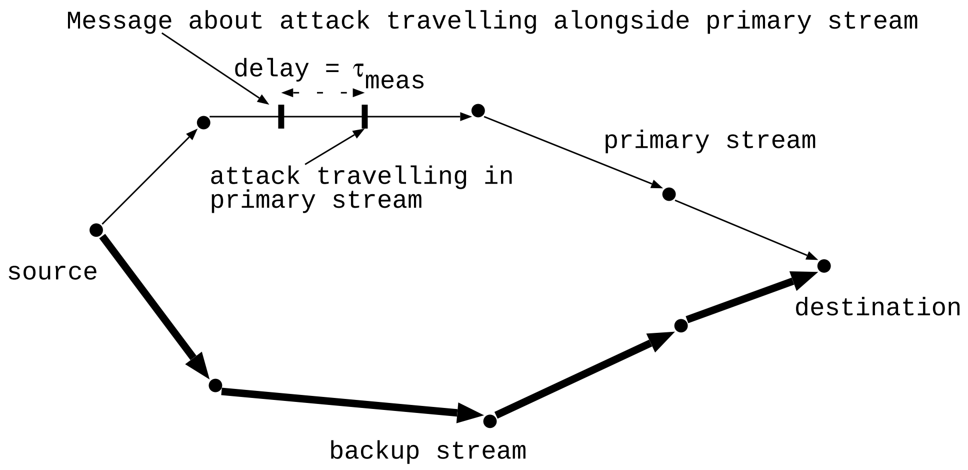

Message about attack travelling alongside primary stream
delay = τ_{meas}
attack travelling in primary stream
primary stream
source
destination
backup stream

Figure 7. Example of automatic protection switching.

very advantageous for scalability of the network. Moreover, having a single delay for all nodes results in simple optical hardware at d, since adapting to different delays on the fly requires significant complexity at the node.

4.2. Application to Loopback

Loopback restoration, in the case of a failure, is performed by the two nodes adjacent to the failure. If node j experiences a failure the data stream is re-routed at node i to travel on the backup channel. Simultaneously, node k receives on the backup stream (see figure 8). This restoration maintains the connectivity of the ring and allows the data to reach the destination despite the failure at j.

Let us consider an attack at node j, as shown in figure 8. Node i is immediately upstream of node j, the source of the attack. Node k is immediately downstream of j. The attack may spread so that the node upstream of the source of the attack will detect an attack while it will not be attacked directly. If we were to detect attacks as failures, loopback might not offer recovery from attack as discussed in section 1.2.

We apply our algorithm to loopback in the following way. In the event of an attack each node attempts to determine whether it is immediately upstream or immediately downstream of the attacked node. In the example in figure 8, node i finds that node j is the source of an attack (by monitoring upstream messages) and re-routes. Node k finds that node j is the source of the attack and re-routes. All other nodes find that they are not immediately upstream or downstream of the attack. Thus, these nodes do not re-route despite the detected attack.

We will use the attack localization algorithm with wait time $T^{wait2} = 2 * max(T_{ij}) + max_i(\tau_i^{proc})$ which gives the node time to monitor backward messages. Messages will consist of the couple $\langle s, flag \rangle$ where s is the status of the node (one of $O.K.$ or $Attack$), and $flag \in \{DontKnow, Mine, NotMine\}$. The flags indicate whether the transmitting node is responsible for the fault or not, or that the node does not yet know if it is responsible for the fault. For this case we will remove messages from the message stream when they are processed. The response function R is shown in figure 9.

Step 1 of the response function R checks incoming messages. If there are none it simply posts the status message with a $DontKnow$ flag $\langle s, DontKnow \rangle$. Step 4 localizes the source of the attack. It posts the flag $Mine$ when no node upstream detects an attack. Step 5 re-routes the node immediately downstream of the attacked node when an $\langle Attack, Mine \rangle$ message is received from the source node. Step 6 re-routes the node immediately upstream of the attacked node when it receives an $\langle Attack, flag \rangle$ message. The upstream nodes need not wait for an $\langle Attack, Mine \rangle$ message because the attack does not propagate upstream.

Let us trace through the messages posted at nodes i, j, k and l, the node downstream from k, when an attack occurs at node j (see table 1). For simplicity let us assume that all measurement are negligibly small and all transmission times are equal so we can examine the nodes at discrete timesteps. Let the attack at node j occur at time t. At this time only node j detects an attack. At time $t + 1$ node j receives an $O.K.$ message from node i and finds it is the source (step 4). Node k detects an attack and receives an attack message from j indicating that it is not the source (step 4). Node i receives the attack message from node j and re-routes (step 6). At time $t + 2$ node k finds that node

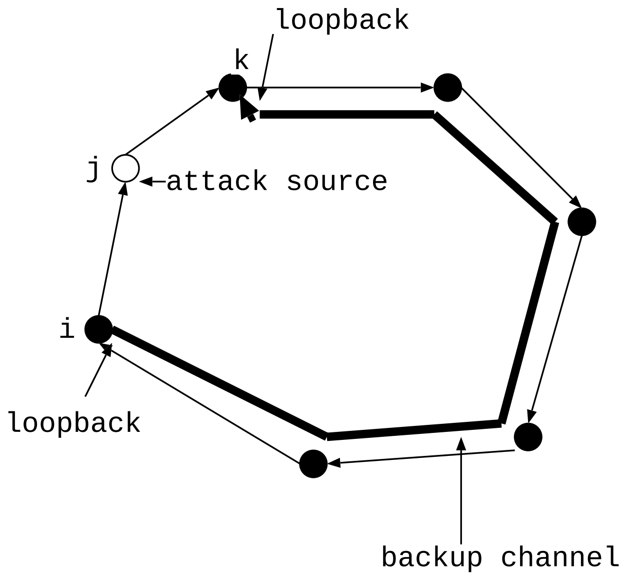


Figure 8. Example of loopback protection.

time	*node i*	*node j*	*node k*	*node l*
t	$\langle O.K., DontKnow\rangle$	$\langle Attack, DontKnow\rangle$	$\langle O.K., DontKnow\rangle$	$\langle O.K., DontKnow\rangle$
$t+1$	$\langle O.K., NotMine\rangle$ *	$\langle Attack, Mine\rangle$	$\langle Attack, NotMine\rangle$	$\langle O.K., DontKnow\rangle$
$t+2$	$\langle O.K., NotMine\rangle$ *	$\langle Attack, Mine\rangle$	$\langle Attack, NotMine\rangle$ *	$\langle Attack, NotMine\rangle$
$t+3$	$\langle O.K., NotMine\rangle$ *	$\langle Attack, Mine\rangle$	$\langle Attack, NotMine\rangle$ *	$\langle Attack, NotMine\rangle$

Table 1. Messages and side-effects of operating R in algorithm 7 when an attack occurs at time t at node j. A decision to re-route is indicated with a *.

j is the source and node k is the next node downstream and re-routes (step 5). Node l also detects the attack at time $t+2$ and receives the $\langle Attack, NotMine\rangle$ message from node k, thereby finding that it is not the source. Since the message indicates that node k is not the source, node l does not re-route.

The timing issues are important, since i and k act independently. If k performs loopback before the traffic on the backup channel has reached k, there will simply be a delay in restoration but no data will be lost. Loopback could fail, however, if k performs loopback *after* the loopback traffic from i has arrived at k. There could be loss of data on the backup channel upon arrival at k. We show that this eventuality cannot occur. Let t be the time at which the attack hits j. At time $t + max\left(\tau_i^{meas}, \tau_j^{meas}\right) + \tau_j^{proc}$, j will send a message to i informing it that the source of the attack is at j. Node i will receive the message that j is the source of the attack at time $t + max\left(\tau_i^{meas}, \tau_j^{meas}\right) + T_{ij}$ and will finish processing the message τ_i^{proc} later. If it takes i τ_i^{loop} time to perform loopback, then i will perform loopback at time $t + max\left(\tau_i^{meas}, \tau_j^{meas}\right) + \tau_j^{proc} + \tau_i^{proc} + T_{ij} + \tau_i^{loop}$. Node k will know that it is not the source of the attack at time $t + max\left(\tau_j^{meas}, \tau_k^{meas}\right) + \tau_k^{proc} + +T_{jk}$. However, the information that is needed by k is whether or not j is the source of the attack. Node k will know that j is the source of the attack and perform loopback at time $t + max\left(\tau_i^{meas}, \tau_j^{meas}\right) + \tau_j^{proc} + T_{jk} + \tau_k^{proc} + \tau_k^{loop}$. We may assume that all the τ^{loop} are equal, all of the τ^{meas} are equal and all of the τ^{proc} are equal. Such an assumption made be made w.l.o.g. because we could take the maximum of all these τs and delay the others to match the maximum. Let us assume, as would be the case in AONs, that transmission delays are proportional to length. From elementary geometry, we know that $|T_{ij} - T_{jk}| \leq$ transmission time from i to k. Therefore, no traffic from i to k placed on the backup channel by loopback will arrive at k before k has performed loopback.

Algorithm 7 R **for loopback**
Input: s, $InMessages$

```
Step 1:  If $InMessages = \emptyset$
Step 2:  then return $M(\overrightarrow{i,j}) = \langle s, DontKnow \rangle$ and $M(\overleftarrow{j,i}) = \langle s, DontKnow \rangle$
         else
Step 3:      Let $DownstreamMessages = \{m \in InMessages$ s.t. $m = M(\overrightarrow{j,i})\}$
             Let $UpstreamMessages = \{m \in InMessages$ s.t. $m = M(\overleftarrow{i,j})\}$
             if $s = Attack$ then
Step 4:          If $\exists M(\overrightarrow{j,i}) \in DownstreamMessages$ s.t. $M(\overrightarrow{j,i}) = \langle Attack, flag \rangle$
                 then return $M(\overrightarrow{i,j}) = \langle Attack, NotMine \rangle$
                 else return $M(\overrightarrow{i,j}) = \langle Attack, Mine \rangle$
Step 5:          If $\exists_{M(\overrightarrow{j,i}) \in DownstreamMessages}$ s.t. $M(\overrightarrow{j,i}) = \langle Attack, Mine \rangle$
                 then Receive from alternate route.
                     return $M(\overrightarrow{i,j}) = \langle Attack, NotMine \rangle$
Step 6:      else    If $\exists_{M(\overleftarrow{i,j}) \in UpstreamMessages}$ s.t. $M(\overleftarrow{j,i}) = \langle Attack, flag \rangle$
                 then Transmit data on alternate route.
                 return $M(\overleftarrow{i,j}) = \langle O.K., NotMine \rangle$
```

Figure 9. Response function, R, for loopback

5. Conclusions and Future Research

We have presented and analyzed an algorithm for attack localization. This algorithm is particularly well suited to the problem of attack propagation which arises in AONs, and by its design satisfies our localization and robustness goal. We applied our algorithm to two common network restoration paradigms, automatic protection switching and loopback. For both paradigms our algorithm guarantees proper network operation, no data loss and bounded delay time regardless of the location of the attack or the physical span of the network.

The algorithm is distributed and its associated delays do not depend on the number of nodes in the network. Hence this algorithm avoids the computational complexity inherent to centralized approaches. It thus achieves the scalability and speed goal. To be accurate we must point out that when we have downstream and upstream messages the one link transition delay is a factor, therefore the algorithm may not scale with one-hop node distances.

Moreover, the delays in attack detection do not depend on the transmission delays in the network. The network management system can therefore offer hard upper-bounds on the loss of data due to failures or attacks. Fault localization with centralized algorithms depends on transmission delays, which are proportional to the distance traversed by the data. Since our algorithm has no such dependence it is equally applicable to local area networks, metropolitan area networks, or wide area networks.

There are several open directions for future research. Subtle forms of attacks can cause cumulative data degradation through a network. Any single degradation may not be severe enough to force an alarm, but the cumulative effect of such degradation over the network may result in invalid data. Extending our approach to detecting cumulative degradation remains an open problem. Another area of interest is the evaluation of reliability afforded by our algorithm for specific fault detection methods at the nodes, explicitly taking false positive and false negative probabilities into account..

References

[1] R.A. Barry *et al.*, “All-Optical Network Consortium - Ultrafast TDM Networks”, *IEEE Journal on Selected Areas in Communications*, vol. 14, no. 5, June 1996, pp. 999–1013.

[2] R. Bhandari, “Optimal Diverse Routing in Telecommunication Fiber Networks”, in *IEEE INFOCOM’94*, vol. 3, pp. 11c.3.1–11.c.3.11.

[3] M. Bischoff, M.H. Huber, O. Jahreis, S.A.F. Derr, F. Ulm, "Operation and Maintenance for an All Optical Transport Network," *IEEE Communications Magazine*. November 1996, pp. 136-142.

[4] C.A. Brackett *et al.*, "A Scaleable Multiwavelength Multihop Optical Network: A Proposal for Research on All-optical Networks", *Journal of Lightwave Technology*, vol. 11, no. 5/6, June 1993, pp. 739–.

[5] G.-K. Chang *et al.*, "Multiwavelength Reconfigurable WDM/ATM/SONET Network Testbed", *Journal of Lightwave Technology*, vol. 14, no. 6, June 1996, pp.1320–40.

[6] A.T. Dahbura, "An efficient algorithm for probabilistic fault diagnosis", *Proceedings of the Nineteenth Annual Conference on Information Sciences and Systems*, 1983, Johns Hopkins University, pg. 89.

[7] A.T. Dabhura, G.M. Masson, "An Efficient Diagnosis Algorithm for t-Diagnosable Systems", *Proceedings of the Seventeenth Annual Conference on Information Sciences and Systems*, 1983, Johns Hopkins University, pp. 456–459.

[8] A.T. Dabhura, G.M. Masson, "An $O\left(n^{2.5}\right)$ fault identification algorithm for diagnosable systems", *IEEE Transactions on Computers*, vol. C-33, June 1984, pp. 486–492.

[9] R.H. Eng, A.A. Lazar, W. Wang, "A Probabilistic Approach to Fault Diagnosis in Linear Lightwave Networks," *IEEE Journal on Selected Areas in Communications*, vol. 11, no. 9, December 1993, pp. 1438-1448.

[10] S.G. Finn, R.A. Barry, "Optical Services in Future Broadband Networks", *IEEE Network*, November/December 1996, vol. 10, no. 6, pp. 7–13.

[11] S.G. Finn, M. Médard, R.A. Barry, "A Novel Approach to Automatic Protection Switching Using Trees", in *Proceedings of ICC'97*.

[12] L.M. Gardner, M. Heydri, J. Shah, I.H. Sudborough, I.G. Trollis, C. Xia, "Techniques for Finding Ring Covers in Survivable Networks" in *Proceedings of GLOBECOM'94*, vol. 3, pp. 1862–1866.

[13] R.W. Haddad, G.M. Masson, "Fault Diagnosis by Linear Programming", *Proceedings of the Seventeenth Annual Conference on Information Sciences and Systems*, 1983, Johns Hopkins University, pp. 460–463.

[14] S.L. Hakimi, A.T. Amin, "Characterization of the connection assignment of diagnosable systems", *IEEE Transactions on Computers*, vol. C-23, January 1974, pp. 86–88.

[15] B.R. Hemenway *et al.*, "Demonstration of a re-configurable wavelength-routed nework at 1.14 terabits-per-second", *OFC 97*, Postdeadline Paper PD-26, February 1997.

[16] G.R. Hill *et al.*, "A Transport Network Layer Based on Optical Network Elements", Journal of Lightwave Technology, vol. 11, no. 5/6, June 1993, pp. 667-679.

[17] M.N. Huber, O. Jahreis, "Supervision and protection concepts for an optical network", *in Proceedings of OFC 95*, pp. 167–168.

[18] Itai, Rodeh

[19] I.P. Kaminow *et al.*, "A Precompetitive Consortium on Wide-band All Optical Networks", *IEEE Journal on Selected Areas in Communications*, vol. 14, no. 5, June 1996, pp. 780–799

[20] I. Katzela, G. Ellinas, T.E. Stern, "Fault Diagnosis in the Linear Lightwave Networks," *IEEE/LEOS 1995 Digest of the LEOS Summer Topical Meetings*, August 1995, pp. 41-42.

[21] I. Katzela amd M. Schwartz, "Schemes for Fault Identification in Communication Networks," *IEEE/ACM Transactions on Networking*, vol. 3, no. 6, December 1995, pp. 753-764.

[22] V.P. Krothapali, K. Nakajima, "On fault identification in t-fault diagnosable analog systems", *Proceedings of the Seventeenth Annual Conference on Information Sciences and Systems*, 1983, Johns Hopkins University, pp. 450–454.

[23] V.P. Krothapali, K. Nakajima, "On adaptive fault identification for certain classes of diagnosable systems", *Proceedings of the Seventeenth Annual Conference on Information Sciences and Systems*, 1983, Johns Hopkins University, pp. 443–449.

[24] C.-S. Li, R. Ramaswami, "Automatic Fault Detection, Isolation and Recovery in Transparent All-Optical Networks", *Journal of Lightwave Technology*, October 1997, vol. 15, no. 10, pp. 1784–1793.

[25] M.W. Maeda, "Management of WDM Optical Networks," *Optical Fiber Communication Conference 97 Tutorial*, February 1997, pp. 141-166.

[26] S.N. Maheshwari, S.L. Hakimi, "On models for diagnosable systems and probabilistic fault diagnosis", *IEEE Transactions on Computers*, vol. C-25, March 1976, pp. 228–236.

[27] P. Mateti, N. Deo, “On Algorithms for Enumerating All Circuits of a Graph”, *SIAM J. Comput.*, vol. 5, no. 1, March 1976.

[28] M. Médard, S.G. Finn, R.A. Barry, “Guaranteed Loopback Recovery over Arbitrary Redundant Graphs”, submitted to *OFC 98*.

[29] M. Médard, S.G.Finn, R.A. Barry, “Automatic Protection Switching for Multicasting in Optical Mesh Networks”, in *Proceedings of Optical Fiber Communication Conference 97*.

[30] M. Médard, D. Marquis, R.A. Barry, S.G. Finn, “Security Issues in All-Optical Networks”, *IEEE Network*, May/June 1997, vol. 11, no. 3, pp. 42–48.

[31] K. Menger, “Zur allgemeinen Kurventheorie”, *Fundamenta Mathematicae*, 10:96–115, 1927.

[32] G.G.L. Meyer, “The PMC system level fault model:maximality properties of the implied faulty set”, *Proceedings of the Seventeenth Annual Conference on Information Sciences and Systems*, 1983, Johns Hopkins University, pp. 443–449.

[33] T. Nishida, S. Hasegawa, A. Kanesama, “Architectural Model for SONET End-to-End Management with Object-oriented Approach,” *IEEE Global Telecommunications Conference* 1989, pp. 1500-1505.

[34] Y. Peng , J.A. Reggia, “A Probabilistic Causal Model for Diagnostic Problem Solving - Part I: Integrating Symbolic Causal Inference with Numeric Probabilistic Inference,” *IEEE Transactions on Systems, Man, and Cybernetics*,” vol. SMC-17, no. 2, March/April 1987, pp. 146-162.

[35] Y. Peng , J.A. Reggia, “A Probabilistic Causal Model for Diagnostic Problem Solving Part II: Diagnostic Strategy,” *IEEE Transactions on Systems, Man, and Cybernetics*,” vol. SMC-17, no. 3, May/June 1987, pp. 395-406.

[36] M. Piepers, “Management of a Fibre Transmission System for Video, Audio and Auxiliary Data Signals,” *IEEE International Broadcasting Convention*, September 12-16, 1996, pp. 414-418.

[37] F.P. Preparata, G. Metze, R.T. Chien, “On the connection assignment problem of diagnosable systems”, *IEEE Transactions on Electronic Computing*, vol. EC-16, December 1967, pp. 848–854.

[38] N. Schroff, M. Schwartz. “Fault Detection/Identification in the Linear Lightwave Network,” Columbia University, 1991.

[39] S.Z. Shaikh, “Span-Disjoint Paths for Physical Diversity in Networks”, in *Proceedings of the IEEE Symposium on Computers and Communications*, 1995, pp. 127–133.

[40] J. Shi, J. Fonseka, “Interconnection of Self-Healing Rings”, in *Proceedings of ICC’96*.

[41] C.-C. Shyur, S.-H, Tsao, Y.-M. Wu, “Survivable Network Planning Methods and Tools in Taiwan”, *IEEE Communications Magazine*, September 1995.

[42] C.-C. Shyur, Y.-M. Wu, C.-H. Chen, “A Capacity Comparison for SONET Self-Healing Ring Networks”, in *Proceedings of GLOBECOM’93*, vol. 3, pp. 1574–1578.

[43] J.B. Slevinsky, W.D. Grover, M.H. MacGregor, “An Algorithm for Survivable Network Design Employing Multiple Self-Healing Rings”, in *Proceedings of GLOBECOM’93*, vol. 3, pp. 1568–1573.

[44] M. Stoer, *Design of Survivable Networks*, Springer-Verlag, 1992.

[45] J.W. Suurballe, “Disjoint Paths in a Network”, *Networks*, 1974, pp. 125–145.

[46] Y. Tada, Y. Kobayashi, Y. Yamabayashi, S. Matsuoka, K. Hagimoto, “OA & M Framework for Multiwavelength Photonic Transport Networks”, *IEEE Selected Areas in Communications*, vol. 14, no. 5, June 1996.

[47] M. Tomizawa, Y. Yamabayashi, N. Kawase, Y. Kobayashi, “Self-healing algorithm for logical mesh connection on ring networks”, *Electronics letters*, 15^{th} September 1994, vol. 30, no. 19, pp. 1615–1616.

[48] P.K. Varshney, C.R.P. Hartmann, J.M. de Faria, Jr., “Application of Information Theory to Sequential Fault Diagnosis”, *IEEE Transactions on Computers*, vol. C-31, February 1982, pp. 164–170.

[49] P.K. Varshney, C.R.P. Hartmann, “Sequential Fault Diagnosis of Modular Systems”, *Proceedings of the 1982 Conference on Information Sciences and Systems*, Princeton, pg. 377.

[50] R.S. Vodhanel *et al.*, “National-scale WDM networking demonstration by the MONET consortium”, *Optical Fiber Communication Conference*, Postdeadline Paper PD-27, February 1997.

[51] O.J. Wasem, “Optimal Topologies for Survivable Fiber Optic Networks Using SONET Self-healing Rings”, in *Proceedings of GLOBECOM’91*, vol. 3, pp. 57.5.1–57.5.7.

[52] //www.ll.mit.edu/aon/

[53] J.S. Whalen, J. Kenney, “Finding Maximal Link Disjoint Paths in a Multigraph”, in *Proceedings of GLOBECOM ’90*, pp. 403.6.1-403.6.5.

[54] T.-H. Wu, “A Passive Protected Self-Healing Mesh Network Architecture and Applications”, *IEEE/ACM Transactions on Networking*, vol. 2, no. 1, February 1994, pp. 40–52.

[55] T.-H. Wu, *Fiber Network Service Survivability*, Artech House, 1992.

[56] T.-H. Wu, M.E. Burrows, “Feasibility Study of a High-Speed SONET Self-Healing Ring Architecture in Future Interoffice Networks”, *IEEE Communications Magazine*, November 1990, pp. 33–51.

[57] T.-H. Wu, R.H. Caldwell, M. Boyden, “A Multi-Period Design Model for Survivable Network Architecture Selection for SDH/SONET Interoffice Networks”, *IEEE Transactions on Reliability*, vol. 40, no. 4, October 1991, pp. 417–432.

[58] T.-H. Wu, W.I. Way, “A Novel Passive Protected SONET Bidirectional Self-Healing Ring Architecture”, *IEEE Journal of Lightwave Technology*, vol. 10, no. 9, September 1992.

[59] W.T. Zaumen, J.J. Garcia-Luna Aceves, “Dynamics of Distributed Shortest-Path Routing Algorithms”, *Proceedings of the 21$^s t$ SIGCOMM Conference*, September 3-6, 1991, in *Computer Communications Review*, vol. 21, no. 4, ACM Press, pp. 31–43.