Adaptive Security-Aware Scheduling Using Multi-Agent System

Maen Saleh
Department of Electrical and Computer Engineering
Western Michigan University, Kalamazoo, MI 49008

Liang Dong
Department of Electrical and Computer Engineering
Baylor University, Waco, TX 76798

***Abstract*—Internet protocol security (IPsec) provides real-time IP packets with confidentiality security service, making them robust against snooping security threats. Conventionally, the security level provided by such protocol cannot be modified according to the status of the network. In this paper, we propose a security-aware scheduling algorithm for a heterogeneous packet switched network. It provides real-time packet flows with guaranteed quality of service (QoS) while adaptively controls the packet's confidentiality security service level. The proposed scheme is modeled using the object-oriented multi-agent methodology. By applying a buffer estimation technique at the network's end nodes, the algorithm provides a real-time network congestion control hence satisfying the network performance metrics. In order to minimize the overhead of network association performed by the IPsec, the algorithm overloads the priority code point fields of the IEEE 802.1Q tagged frame format. This approach helps meet both QoS and security requirements for real-time data flows.**

I. Introduction

Data communication network evolves to provide better real-time services through the Internet. Real-time scheduling algorithms can be implemented to guarantee the required quality of service (QoS) for different classes of data streams in the network. Besides guaranteeing the QoS of real-time data flows, network technologies should be capable of providing such flows with a secure data transition [1]. This can be achieved by applying appropriate security services on real-time applications, hence protecting them from various network security threats [2]. In order to obtain the security services, various security protocols can be implemented such as the secure socket layer (SSL) protocol, the transport layer security (TLS) protocol, and the Internet protocol security (IPsec) protocol. However, with these protocols, any dynamic change in the network cannot affect the provided security level. Therefore, the pre-defined network performance metrics (NPMs) are not taken into account and the QoS may not be guaranteed. This can lead to a catastrophe, especially for some hard-deadline real-time applications. Implementing a real-time scheduler independently from any security protocol does not meet the network requirements either [3]. In order to model such complex secure real-time network, a real-time multi-agent system model needs to be used. The whole system is modeled by interactive entities that cooperate with a time-constrained protocol [4].

In this paper, we propose an object-oriented multi-agent system that provides security-aware scheduling for a real-time packet switched network. The proposed algorithm implements a network monitoring technique that provides buffer estimation at the network's edge (end nodes). It provides a real-time network congestion control and protects the network from being congested by heavy traffic load. We overload the priority code point (PCP) fields of the IEEE 802.1Q tagged frame format, where the new value represents the code associated with the confidentiality security algorithm. In doing so, we minimize the overhead of the security association (SA) performed by the security protocol. With the proposed algorithm, the confidentiality security level of the data streams can be adapted through the congestion control mechanism without affecting the NPMs. Therefore, the QoS can be guaranteed for all data traffics.

II. Adaptive CONF/Diff-EDF Security-Aware Scheduler

Our system is designed based on the object-oriented multi-agent system. It combines the differentiated earliest-deadline-first (Diff-EDF) scheduler with a confidentiality security service upgrading unit. We refer to this architecture as adaptive CONF/Diff-EDF. Decomposition, modeling, and communication protocol are the three main design phases for the proposed system. For the decomposition, the architecture is decomposed into three main heterogeneous cooperating entities: source, destination, and edge router. The edge router includes four sub-entities: coordinator, server, Diff-EDF scheduler, and buffer queue. At the modeling phase, each entity is modeled by an interactive agent through defining its main sub-tasks and behaviors. The communication protocol governs both interactions and communication schemes between the real-time agents. Fig. 1 shows the multi-agent system model.

A. Source Agent

The source agent is the data packet generator. It generates one of the two types of real-time data packets: video or audio. Each packet has a fixed size $P_s = 1.46$ KB (1500 bytes), which is the maximum size of the Ethernet packet frame. The source agent sends real-time traffic f with a rate of λ_f. An exponential distribution with mean $1/\lambda_f$ is used to for the packet inter-arrival time. Another exponential distribution with mean $1/\mu_f$ is used for the packet service time, where μ_f is the service rate given by $\mu_f = 8B_w/P_s$, where B_w is the average aggregate bandwidth needed for both types of real-time traffic

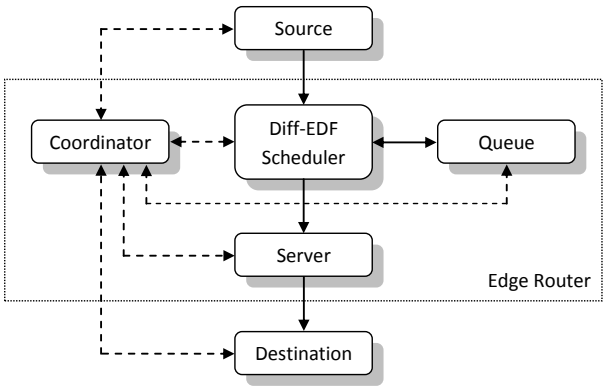

Fig. 1. Multi-agent system model.

(video and audio). A uniform distribution is used to generate the relative deadline D_f associated with real-time traffic f. A QoS requirement is specified for traffic f in terms of deadline miss rate Φ_f.

In order to combat the unauthorized access of date packets (snooping threat), the source agent uses cryptographic security algorithms. A confidentiality security service level, ranging from 1 to 8, is applied to each real-time packet, which indicates one of the eight cryptographic algorithms used by the source agent. Index 1 indicates the weakest cryptographic algorithm, while index 8 indicates the strongest one. Table I shows these security algorithms, which are based on a study performed on 175 MHz processor machine [5]. In Table I, μ_{sj} is the data rate in KB/ms that can be enhanced using the jth cryptographic security algorithm, and S_{lj} is a number between 0.08 and 1, which indicates the efficiency of the security algorithm with respect to the strongest algorithm $S_{lj} = 13.5/\mu_{sj}$. The source agent interacts with the coordinator agent by sending requests to serve its real-time traffic with QoS requirements.

B. Destination Agent

The destination agent performs a first-come first-served (FCFS) scheduling algorithm on the received packets from the server agent. It sends two parameters to the coordinator agent: 1) Processing rate P_{ki}, which is sent at the initiation process; 2) The size of its available buffer B_{kj}, which is sent every time period T specified by the coordinator agent. Such information is used by the coordinator agent for the confidentiality security enhancement process.

Index (j)	Sec. Algorithm	S_{ij}	μ_{sj} (KB/ms)
1	SEAL	0.08	168.75
2	RC4	0.14	96.43
3	Blowfish	0.36	37.5
4	Knufu/Khafre	0.40	33.75
5	RC5	0.46	29.35
6	Rijndael	0.64	21.09
7	DES	0.90	15
8	IDEA	1.00	13.5

TABLE I
CRYPTOGRAPHIC ALGORITHMS

C. Edge-Router's Sub-Agents

Due to its adaptive feature, the proposed real-time scheduling with security awareness cannot be modeled based on the conventional static queuing theory. Upon receiving a request from the source, the coordinator models the scheduling process as a general Brownian motion with negative motion drift parameter $(-\theta)$ [6], such that

$$\theta = \frac{2(1-I)}{\sum_{f=1}^{N} \lambda_f (I_f^2 \sigma_{1f}^2 + \sigma_{2f}^2)} \tag{1}$$

where σ_{1f} is the standard deviation of the inter-arrival time for flow f, σ_{2f} is the standard deviation of the service time for flow f. The intensity of traffic flow f is $I_f = \lambda_f/\mu_f$, the total intensity of all flows is $I = \sum_{f=1}^{N} I_f$, and N is the number of flows. With the smallest deadline miss rate of all flows being Φ_{min}, the coordinator obtains the parameter C_f as

$$C_f = \theta^{-1} \log(\Phi_f/\Phi_{\text{min}}) \tag{2}$$

where Φ_f is the required deadline miss rate of traffic f. The coordinator evaluates the feasibility of serving such request by estimating the flow deadline miss rate

$$\hat{\Phi}_f = \exp(-\theta(D_{\text{avg}} - C_f)) \tag{3}$$

where D_{avg} is the average effective deadline for all data flows, such that $D_{\text{avg}} = \sum_{f=1}^{N} I_f(\Phi_f + C_f)$.

If the estimated deadline miss rate meets the QoS requirement, i.e. $\hat{\Phi}_f < \Phi_f$, the coordinator performs the following actions: (1) Sending an acceptance message to the source; (2) Passing the parameter C_f to the Diff-EDF scheduler; and (3) Passing the required deadline miss rate Φ_f to the server. The source agent responds to the acceptance message by sending its real-time data packets to the scheduler. The scheduler performs a shadow function to obtain the packet's effective deadline D_{ef}, such that $D_{ef} = D_f + C_f$. The scheduler then forwards the packet to the queue agent, which queues the packet based on its effective deadline.

The functionality of the server agent is to complete the process of serving the real-time packets. Once the sever completes serving a current real-time packet, it interacts with the coordinator by sending an idle status message. The coordinator responds by interacting with the queue agent to perform a dequeing process. The queue agent fetches a packet that is closest to expire (with smallest D_{ef}) and forwards it to the scheduler. The scheduler passes the packet to the server. Once it receives the packet, the server performs the following actions: (1) Changing its current status to busy; (2) Serving the unexpired packet or dropping the expired packet; (3) Forwarding the packet to its destination according to its MAC address; and (4) Keeping track of two counters: n_f the number of packets served for the destination of traffic flow f and t_f the sum of time differences of served packets for the destination of traffic f, i.e. $t_f = \sum_{i=1}^{n_f}(t_i - t_{i-1})$.

Over every time period T, the coordinator interacts with both server and destination agents requesting for the server's counter information (n_f, t_f) and the destination's resource

information (B_f, P_f). Upon receiving such information, the coordinator finds the mean inter-arrival time $(1/\zeta_f)$ for the packets of traffic flow f delivered to their destination, where

$$1/\zeta_f = t_f/(n_f - 1). \tag{4}$$

In its data base, the coordinator stores the information about different cryptographic algorithms that are used for enhancing the packet's confidentiality security service level (Table I). Based on this, the coordinator determines the length of buffer L_{fj} that is needed to enhance n_f real-time packets with the jth algorithm, such that $L_{fj} = \mathcal{P}_f \zeta_f$, where $\mathcal{P}_f$ is the total processing time of the packets of traffic flow f. It takes into account two delays: the delay of solving the problem of two or more equally prioritized packets ($D_{f,\text{equal\_priority}}$) and the delay of the preemption process when an arriving packet is closer to expiration than the remaining time of the currently processing packet ($D_{f,\text{preemption}}$). We have

$$\mathcal{P}_f = D_{f,\text{equal\_priority}} + D_{f,\text{preemption}} + \tau_{fj} \tag{5}$$

where τ_{fj} is the time required to decrypt a packet of 1500 bytes (1.46 KB) using the jth security algorithm. $\tau_{fj} = 1.46 \text{ KB}/(\mu_j \beta)$, and β is the processing rate factor $\beta = P_f/175$ MHz.

The coordinator compares the length of available buffers at the destination with the required length of buffers to enhance n_f packets using different confidentiality algorithms. It adopts the strongest security algorithm with no congestion in the network. Given that the length of available buffers at the destination of traffic f is B_f and the length of buffers needed to enhance n_f packets to security level z is L_{fz}, the coordinator enhances/reduces security to level z, or stays at the same security level z such that

$$L_{fz} \leq B_f < L_{f(z+1)}. \tag{6}$$

Once the decision on security level is made, the coordinator notifies the source. No notification will be sent if the decision is to stay at the same security level. Upon receiving a notification, the source agent applies corresponding new security enhancement algorithm to the packets to be sent.

III. Feedback IPsec and IEEE 802.1Q PCP Overloading

IPsec is a network-layer protocol that is implemented to provide the required security services to all IP based applications. In order to establish a secure data channel between two parties, IPsec performs a security association phase, where the required security parameters are initialized such as security service's algorithm, initialization data, encryption data keys, and security modes. Since our data unit is a real-time data packet, IPsec protocol seems to be suitable for the development.

According to the IPsec protocol, every time we change the security algorithm on the data packets, we have to terminate the pre-initiated secure session and then negotiate on the new security parameters to be used for encryption/decryption processes. This leads to an additional overhead, and thus the overall performance of the network is affected. According to a study performed on two 206 MHz-processor machines, it is found that it requires about 167 ms to complete the handshaking process needed to establish a security association between two hosts [7]. The handshaking process is mainly based on two phases. The first phase is to establish the secure data channel that is needed to exchange the security parameters between the hosts. It is based on the security main mode that implements the Diffie-Hellman key exchange method. The second phase performs the process of exchanging the association parameters through the established secured channel. The calculations ignore the process of deriving the RSA security keys by using a pre-existing shared key between the two hosts, which is similar to our development. In our development, we have implemented a feed-back IPsec protocol with pre-existing shared keys on 175 MHz processor. The protocol receives a feed-back about the status of the network. Accordingly, it changes the security level of the real-time packets. Due to the SA negotiation phase, an additional overhead D_{sec} shall be added to the system with each change in the security level, such that $D_{sec} = P_{ki} \times 167\text{ms}/206\text{MHz}$, where P_{ki} is the processing rate.

For our adaptive security-aware scheduling, a question that appears is: Without a security association performed between the source and destination agents, how does the destination host know the type of security algorithm that was applied by the source host on the real-time data packet? We solve this problem by overloading the PCP fields of the IEEE 802.1Q tagged frame format [8]. The IEEE 802.1Q modifies the Ethernet frame format by adding a 4-byte field between the source MAC address field and the type/length field as shown in Table II. It is implemented to define the functionality of virtual LANs (VLANs), where the one physical Ethernet network is divided into a group of logically shared networks for security aspect. The IEEE 802.1Q tag consists of two main fields. As shown in Table III, the first field is the tag protocol identifier (TPID) with a size of 2 bytes and a value of 0x8100. It is used to identify the beginning of the IEEE 802.1Q tagged frame. The second field is tag control information (TCI) with a size of 2 bytes. This field is divided into three fields: (1) PCP with a size of 3 bits, which is used to give priority for different

Preamble	D. MAC Add.	S. MAC Add.	**IEEE 802.1Q**
8 Bytes	6 Bytes	6 Bytes	**4 Bytes**

Type/Length	Payload	CRC
2 Bytes	n Bytes	4 Bytes

TABLE II
IEEE 802.1Q Tag Insertion

16 Bits	3 Bits	1 Bit	12 Bits
TPID	PCP	CFI	VID
0x8100	000 −111	0	All Zeros

TABLE III
IEEE 802.1Q Fields

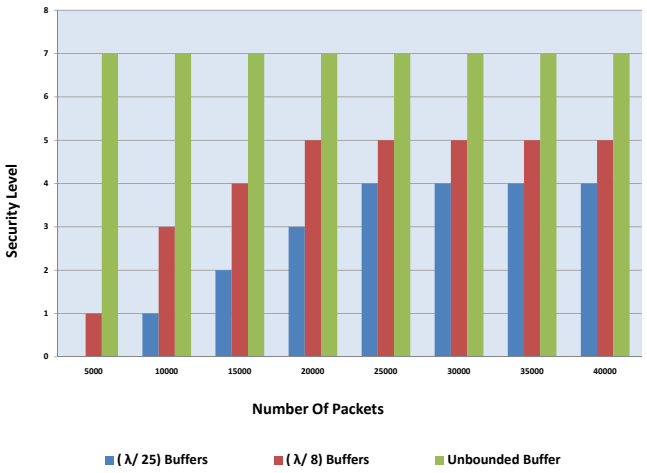

Fig. 2. Buffer effect on security level enhancement.

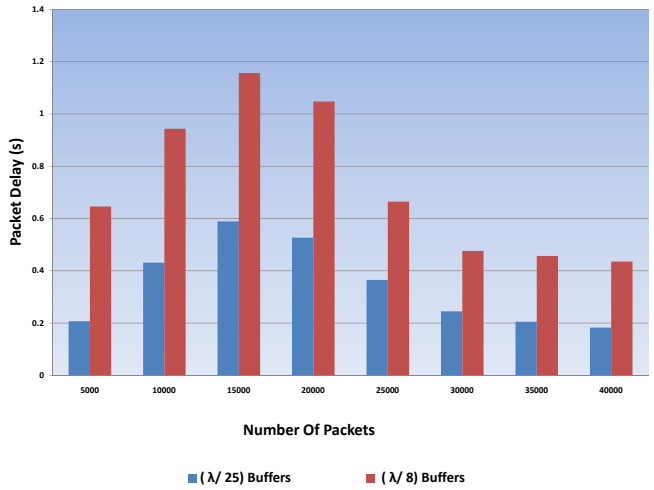

Fig. 3. Buffer effect on average packet delays.

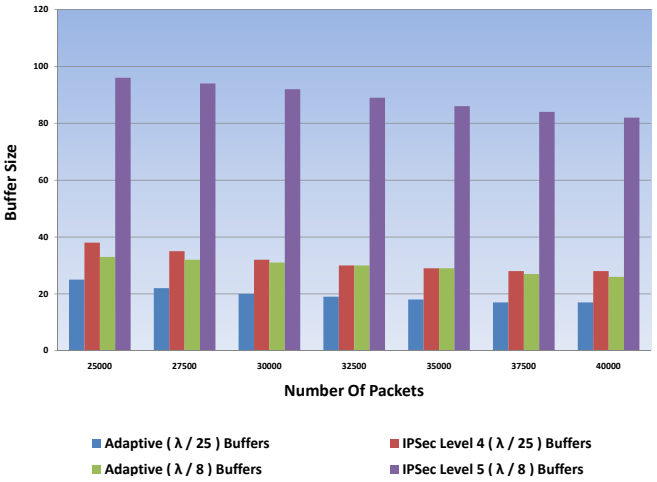

Fig. 4. Average buffer consumption at the destination agent.

data types (text, video, audio, etc.) as shown in Table IV; (2) canonical format indicator (CFI) with a size of 1 bit, which is used to define the format of the MAC address with a 0 meaning that the address is a canonical format (It is always set to 0 for Ethernet frames); (3) virtual LAN identifier (VID) with a size of 12 bits, which specifies the VLAN for which the real-time data belongs.

In our development, the value for the VID was set to all zeros, which means that the IEEE 802.1Q tag is only used for priority aspects. It also indicates that all packets belong to one VLAN. Since the implementation of the PCP field is left to the user, we overload this field to represent the priorities of the cryptographic algorithms that are used in the security enhancement process as shown in the last column of Table IV. Upon receiving the real-time data packet, the destination agent checks the PCP field and determines the cryptographic algorithm that is implemented on the packet. Consequently, by using a hash table for the pre-defined keys, it can decrypt the packet and extract the original payload.

IV. Numerical Results

We simulate a network with N pairs of distinct source and destination, $N = 2, 4, \ldots, 32$. Among the N streams of traffic, there are $N/2$ real-time video streams and $N/2$ real-time audio streams. Each source has a sending rate of $\lambda_f = 1250$ packets per second and starts sending packets with the lowest security level. The edge router will notify each source to update to an appropriate security level. The length of initially available buffers at the destination is either $\lambda_f/25$, $\lambda_f/8$, or λ_f (the unbounded case), multiplied by unit time. Each destination agent has a processing rate factor $\beta = 1$, i.e. a processing speed of 175 MHz. The performance of our proposed system is evaluated from different perspectives:

Network Performance at the Destination Agent: In this simulation, we demonstrate the performance of the proposed algorithm at the destination agent. Fig. 2 and Fig. 3 show the effect of initial buffer length at the destination agent on both the confidentiality security enhancement process and the average total packet delay, respectively. The security service level with larger number of available buffers will be higher, since destinations have more flexibility in decrypting packets. However, this affects the QoS in terms of average total packet delays. Fig. 4 shows the consumption of the destination's buffering system, which contributes to the network congestion. In the figure, we compare the efficiency of our proposed adaptive algorithm with the IPsec protocol at a static security level for the two cases of initial available buffers: $(\lambda_f/25)$ and

PCP	Traffic Type	Traffic Priority	Sec. Algorithm
0	Background	0 (Lowest)	SEAL
1	Best Effort	1	RC4
2	Excellent Effort	2	Blowfish
3	Critical Applications	3	Knufu/Khafre
4	Video, $<$ 100 ms Latency	4	RC5
5	Video, $<$ 10 ms Latency	5	Rijndael
6	Internetwork Control	6	DES
7	Network Control	7 (Highest)	IDEA

TABLE IV
Overloading the IEEE 802.1Q PCP Fields

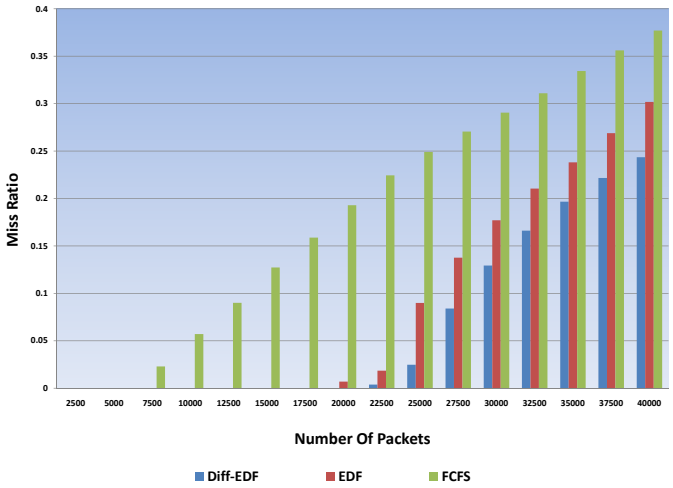

Fig. 5. Miss ratio at the server agent.

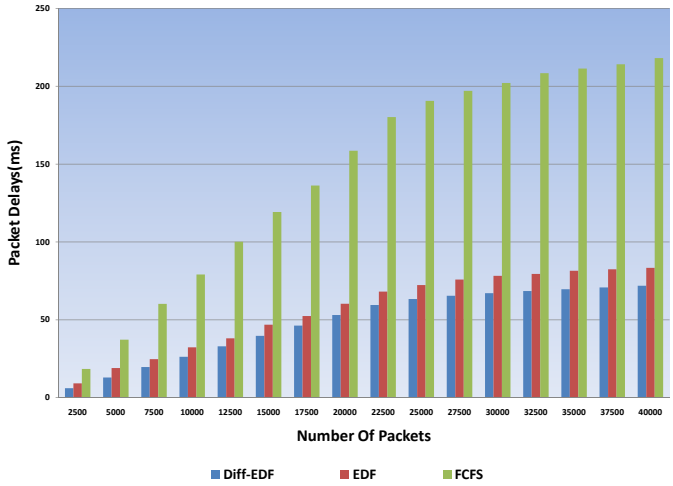

Fig. 6. Average total packet delay at the queue agent.

$(\lambda_f/8)$. The static security level is the steady state security level of the proposed adaptive algorithm (as in Fig. 2). As shown in Fig. 4, the adaptive algorithm protects the destination's buffer from being congested by heavy traffic load.

Network Performance at the Edge Router Agent: In this simulation, we demonstrate the efficiency of using the Diff-EDF algorithm at the scheduler agent over the earliest-deadline-first (EDF) and the first-come-first-serve (FCFS) scheduling algorithms. Fig. 5 and Fig. 6 show two QoS metrics, respectively, the miss ratio (at the server agent) and average total packet delay (at the queue agent). As it shows, the Diff-EDF scheduler has least miss ratio and average total packet delays, therefore it is suitable for our time-critical video/audio packets.

Security Level Changes with a Feedback IPsec: In this simulation, we demonstrate the performance of our proposed adaptive security-aware scheduling algorithm over the implemented feedback IPsec protocol. We measure the number of security level changes of the feedback IPsec protocol at different values of both initial available buffers and the negotiated time interval (T) (Fig. 7). The more changes it

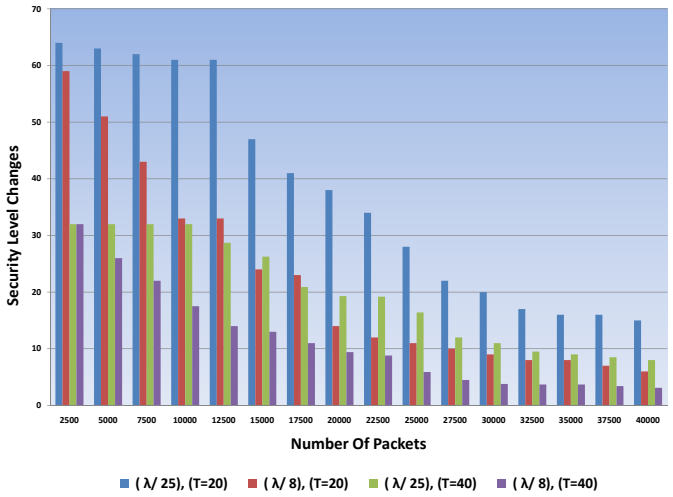

Fig. 7. IPsec security level changes.

makes, which yields to a repeated security association process, the more system overhead it results. The proposed scheme eliminates the repeated SA phase performed by IPsec hence less overhead. It increases the chance to meet the QoS requirements in terms of miss ratio and average total packet delay.

V. Conclusion

Implementing security-aware scheduler becomes a high demand for real-time heterogeneous networks. In this work, we propose a security-aware scheduler based on an object-oriented multi-agent system model. It combines the Diff-EDF scheduler with an adaptive confidentiality enhancement unit. This architecture efficiently utilizes the network's buffering system and regulates the traffic load in the network through implementing a resource estimation methodology. The overhead of the security association performed by the IPsec is minimized by overloading the PCP fields of the IEEE 802.1Q tagged frame format with indexes of security algorithms. The proposed scheme preserves the real-time network performance by meeting both QoS and security requirements.

References

[1] A. Apvrille and M. Pourzandi, "XML distributed security policy for clusters," *Computers & Security*, vol. 23, pp. 649–658, 2004.
[2] E. Cody, R. Sharman, R. H. Rao, and S. Upadhyaya, "Security in grid computing: A review and synthesis," *Decision Support Systems*, vol. 44, pp. 749–764, 2008.
[3] S. H. Son, R. Mukkamala, and R. David, "Integrating security and real-time requirements using covert channel capacity," *IEEE Trans. Knowledge Data Eng.*, vol. 12, no. 6, pp. 865–879, Nov. 2000.
[4] R. Xie, D. Rus, and C. Stein, "Scheduling multi-task multi-agent systems," in *Proc. Int. Conf. Autonomous Agents*, May 2001, pp. 159–160.
[5] T. Xie and X. Qin, "Scheduling security-critical real-time applications on clusters," *IEEE Trans. Comput.*, vol. 55, no. 7, pp. 864–879, Jul. 2006.
[6] H. Zhu, J. P. Lehoczky, J. P. Hansen, and R. Rajkumar, "Diff-EDF: A simple mechanism for differentiated edf service," in *Proc. IEEE Real-Time Embedded Tech. App. Symp.*, 2005, pp. 268–277.
[7] P. G. Argyroudis, R. Verma, H. Tewari, and D. O'Mahony, "Performance analysis of cryptographic protocols on handheld devices," in *Proc. IEEE Int. Sym. Network Computing and Applications*, 2004, pp. 169–174.
[8] K. G. Paterson, "A cryptographic tour of the ipsec standards," *Inf. Secur. Tech. Rep.*, vol. 11, pp. 72–81, Jan. 2006.