ABSTRACT

To secure real-time communication in wireless LANs it is pertinent to implement real time cryptographic services in software or hardware. In this paper we evaluate the use of software based inline encryption algorithms for wireless LANs that are implemented in the Layer Service Provider as defined by WinSock 2 for Windows. The evaluated encryption algorithms run on each PC that is part of the wireless LAN. We present the throughput requirements from the inline encryptors for various multimedia applications such as video conferencing, collaborative work, distributed data bases and distributed processing. Our measurements show that software implementation of various encryptors provides enough throughput as required by the above applications.

I. INTRODUCTION

Wireless Local Area Networks (WLAN) markets are driven by a number of applications or scenarios such as: 1) LAN installation for temporary events in which wiring is very costly or impossible, 2) Extension of existing networks which may be permanent or temporary, 3) Savings in terminals. e.g., patients admission in hospitals in which the process is done in front of the patient without the need to move him to the desk, 4) LAN installation in old buildings in which cost of running cables is high, etc.

In these examples users focus on data applications like email, file transfers and database transactions. Future WLANs will need to support multimedia applications, e.g., teleconferencing, videoconferencing. These applications require guaranteed service, e.g. throughput, maximum time delay and time delay variation. In case these guarantees are not met, the quality and effective delivery of these applications will be severely degraded.

In 1997 the Wireless LAN standard, IEEE 802.11 [10] was approved for both physical layers, Infrared and Radio with employ either Narrowband or Spread Spectrum communication techniques. In this paper we will consider radio WLANs which employ Spread Spectrum (SS) techniques. Spread spectrum techniques provide resistance to intentional jamming by another source and the degrading effects of multi-path transmission. The characteristics of SS modulation are also advantageous from the security standpoint, since both direct sequence (DS) SS and frequency hopping (FH) SS distribute the bits of transmission information for a chip duration. The security aspects of SS communication have been investigated in [11]. The authors concluded that the use of SS as the only security mechanism will not be sufficient. Active intruders in the same service area can easily know or detect the initial spreading code like the paging code and this intrusion can be virtually undetectable [1], [3]. Therefore, the design and implementation of secure communication is of seminal importance in wireless networks.

Moreover, the integration of multimedia applications with security services is even a more challenging task. This is due to the fact that interactive applications (e.g., video conferencing) with strict Quality of Service requirements (in terms of throughput) will compete with the security services on the same scarce network and system resources.

Since the current Wireless LAN technology can sustain data rates of up to 10 Mbps, the WLAN can support only a number of multimedia applications with relatively low bandwidth requirements such as video conferencing, text email, http and multimedia mail/notes. Table 1 shows the candidate applications with their re-

The work was supported in part by NSF under contract number CISE-CDA-9529462.
quired throughput.

Inline security service is defined as a mechanism, implemented either in hardware or software that encrypts/decrypts in real-time every packet transmitted/received by the application. Some military applications use cryptography boxes plugged into the very high speed network lines, e.g., FASTLANE and TACLANE (members of NSA's MISSI security family) [6]. These cryptographic boxes are a necessity in a very high-speed communication network (tens or hundreds of Mbps). Due to their high cost, these boxes are usually shared by a number of computers.

Dedicated cryptography hardware may not be the best choice for WLANS due to the WLANS characteristics: 1) relatively low data rates, 2) relatively low cost, 3) used mostly in portable devices, and 4) a distributed mobile environment.

In this paper we investigate the use of software based inline encryptors which are implemented in the Layer Service Provider as defined by WinSock 2 for Windows. The evaluated algorithms run on each PC that is part of the wireless LAN. Software based inline encryptors are suitable for WLANS characteristics listed above and has the additional advantages: it is cheaper than hardware implementation and more flexible for upgrades. The disadvantage is that its performance is limited by the computer system configuration which impedes the implementation of computational intensive encryption algorithms such as public key cryptography.

It is important to understand that the suitability of using inline software encryptors is determined by the application throughput requirements. To make this determination we have measured the throughput of a number of encryption algorithms which we have implemented in software. The throughput is obtained for two computer system configurations and different loads. We also consider additional delay penalties encountered by a packet in the system caused by other communication protocol layers.

Our measurements show that software implementation of various encryptors provides enough throughput as required by the above applications.

The paper is organized as follows. In the next section we describe the proposed software architecture. The encryption algorithms we have implemented in software and their performance is depicted in Section III. Section III also provides the performance analysis necessary for choosing the proper software based inline encryptors. The conclusions are provided in Section IV.

### II. SOFTWARE IMPLEMENTATION

We implemented our software encryption algorithms using CryptoAPI [4] as crypto service functions in WinSock 2 Layered Service Provider [7] that resides between the principal WinSock 2 DLL and the lower-level base transport provider (see Figure 1).

At the source, each packet that is generated by the application is intercepted at the layer service provided, is encrypted using CryptoAPIs and is sent down to the base transport protocol. At the destination the packets that belong to this application are intercepted by the layer service provider, are decrypted using CryptoAPIs and are forwarded to the application.

The proposed architecture has the following advantages:

- We can use any off-the-shelf applications, i.e., the applications do not need to be modified and made aware of the fact that we use encryption techniques.
- We can use any off-the-shelf network interface card, i.e., we do not need to choose a network interface card that incorporates encryption capability either in hardware or software.
- Easy upgrade of the encryption algorithms.

Of course the proposed implementation may not be suitable for very high speed networks since the encryption throughput may hinder the use of the high speed channel.

### III. EXPERIMENTS

#### A. Encryption Throughput Experimental Results

In this subsection we provide a description of the experimental results, in terms of encryption throughput ($B_{enc}$), we have obtained for a number of encryption algorithms.

<table>
<thead>
<tr>
<th>Types of Applications</th>
<th>Required Throughput</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-Interactive Applications</td>
<td></td>
</tr>
<tr>
<td>text email</td>
<td>2.4 kbps</td>
</tr>
<tr>
<td>http</td>
<td>9.6 kbps</td>
</tr>
<tr>
<td>multimedia mail</td>
<td>&lt; 64 kbps</td>
</tr>
<tr>
<td>multimedia notes</td>
<td>&lt; 100 kbps</td>
</tr>
<tr>
<td>Interactive Multimedia Apps</td>
<td></td>
</tr>
<tr>
<td>telephone-quality audio</td>
<td>64 kbps</td>
</tr>
<tr>
<td>video conferencing</td>
<td>128 kbps - 1 Mbps</td>
</tr>
<tr>
<td>MPEG video</td>
<td>1.5 Mbps</td>
</tr>
</tbody>
</table>

Table 1. Candidate Applications for Multimedia Wireless LANs
We have integrated in our ISLSP a number of bulk encryption algorithms provided by CryptoAPI: RC2 (block cipher), RC4 (stream cipher) which are symmetric-key algorithms which imply that both encryption and decryption operations use the same key. We have also integrated RSA [5] public key algorithm which encrypts and decrypts data much slower than other symmetric algorithms.

We have tested all the above encryption algorithms in the application layer. We measured the encryption throughput which is defined as the number of bytes processed per second. These measurements reported in Table 2 were taken for two computer configurations M1 and M2 as well as different loads. Available resources of CPU and main memory are the main determinants in choosing the experiments for Table 2. Based on output from system management tools of Windows 95/NT, we have determined the range of throughput for each bulk encryption algorithm.

From Table 2 we observe almost identical throughput of symmetric encryption and decryption algorithms. In public key algorithms such as RSA the bottleneck is in the encryption throughput.

### B. Suitability of Software Inline Encryption for Multimedia Wireless LANs

In this subsection we investigate the use of software based inline encryption for WLANs that support multimedia applications. We will take into consideration 1) the wireless LAN characteristics such as throughput, 2) the applications' QoS requirements in terms of throughput, 3) encryption/decryption throughput that is determined by the computer configuration and encryption algorithms (as shown in the previous section), and 4) additional processing overhead incurred by other protocol layers.

We proceed with the following notations:

- $D$ data packet size (bits)
- $B_w$ wireless LAN effective bandwidth (bps), i.e., we take into account the channel noise and competition on the channel from other nodes in the WLAN. Therefore, this effective bandwidth can be significantly less than the WLAN data rate.
- $B_{app}$ bandwidth requested by the application (bps).
- $B_{enc}$ encryption throughput at the source (bps)
- $B_{dec}$ decryption throughput at the destination (bps)
- $T_{SOH}$ the time overhead spent by a packet at the source. This time includes the time required to process a packet in all the protocol layers (physical layer, data link layer, device driver, network and transport layer). This delay also includes any queuing delays in these protocol layers.
- $T_{DOH}$ the time overhead spent by a packet at the destination. This time includes the time required to process a packet and its queuing delay all the protocol layers.
- $B_S$ the maximum source bandwidth (bps), the maximum rate at which the source PC sends data to the network interface card, assuming that there is always a packet available to be sent from the source.
- $B_D$ the maximum destination bandwidth (bps), the maximum rate at which the destination PC sends data to the application, assuming that there is always a packet available to be sent to the destination.
Figure 2. Application-to-Application Packet Journey

from the network:
- the encryption delay \( \frac{D}{B_{\text{enc}}} \)
- the decryption delay \( \frac{D}{B_{\text{dec}}} \)

Figure 2 depicts the application-to-application packet journey using the above notations.

To sustain the end-to-end bandwidth required by an application with packet size \( D \), \( B_{\text{app}} \) we require:

\[
\min(B_S, B_D, B_w) \geq B_{\text{app}} \quad (1)
\]

We now compute the source and destination bandwidths, \( B_S \) and \( B_D \), as a function of the protocols’ overhead and of the encryption and decryption throughput. We compute the time spent by a packet at the source and destination PC. Since at the source and at the destination we have one CPU that handles the protocols in all the communication layers including the encryption/decryption algorithms and assuming high loads, i.e., there is always a packet for transmission from the application layer (worst case computation) we obtain:

\[
\frac{D}{B_S} = \frac{D}{B_{\text{enc}}} + T_{SOH} \quad (2)
\]

\[
\frac{D}{B_D} = \frac{D}{B_{\text{dec}}} + T_{DOH} \quad (3)
\]

Using equations (2) and (3) we obtain:

\[
B_S = \frac{1}{\frac{1}{B_{\text{enc}}} + \frac{T_{SOH}}{D}} \geq B_{\text{app}} \quad (4)
\]

\[
B_D = \frac{1}{\frac{1}{B_{\text{dec}}} + \frac{T_{DOH}}{D}} \geq B_{\text{app}} \quad (5)
\]

The minimal encryption and decryption throughput to sustain the application bandwidth with the given system parameters is given by

\[
B_{\text{enc}}^{\text{min}} = \frac{1}{\frac{1}{B_{\text{app}}} - \frac{T_{SOH}}{D}} \quad (6)
\]

\[
B_{\text{dec}}^{\text{min}} = \frac{1}{\frac{1}{B_{\text{app}}} - \frac{T_{DOH}}{D}} \quad (7)
\]

In this case we can come up with the following decision regarding the use of inline software encryption with throughput \( B_{\text{enc}} \):

Given: \( B_{\text{app}} \), and \( T_{SOH} \)
If \( B_{\text{enc}} \geq B_{\text{enc}}^{\text{min}} \) use this inline software encryptor (Figure 3).

For example: for a videoconferencing session of 1Mbps, wireless LAN bandwidth of 2 Mbps, \( D = 64\text{Bytes} \), and \( T_{SOH} = 510\mu\text{sec} \), the inline encryptor throughput \( B_{\text{enc}} \) has to be at least 256 Mbps.

Using Equation (6) and (7) the maximum application bandwidth, assuming \( B_w \geq B_{\text{app}} \), \( B_{\text{app}}^{\text{max}} \) is given by:

\[
B_{\text{app}}^{\text{max}} = \min[(\frac{1}{\frac{1}{B_{\text{enc}}} + \frac{T_{SOH}}{D}}), (\frac{1}{\frac{1}{B_{\text{dec}}} + \frac{T_{DOH}}{D}})] \quad (8)
\]

For example: for an inline encryptor with bandwidth of 4 Mbps, wireless LAN bandwidth of 2 Mbps, \( D = 64\text{Bytes} \), and \( T_{SOH} = 1\text{msec} \), the inline encryptor can sustain applications of at most 454 Kbps.

For given values of \( B_w, B_{\text{app}}, B_{\text{enc}} \) and \( D \), the maximum value of \( T_{SOH} \), denoted by \( T_{SOH}^{\text{max}} \) that will still provide the required application bandwidth is given by:

\[
T_{SOH}^{\text{max}} = D \times (\frac{1}{B_{\text{app}}} - \frac{1}{B_{\text{enc}}}) \quad (9)
\]

Figure 5 shows \( T_{SOH}^{\text{max}} \) as a function of \( B_{\text{enc}} \) for various values of \( B_{\text{app}} \) and \( D \). Figure 4 shows \( B_{\text{app}}^{\text{max}} \) as a function of \( B_{\text{enc}} \) for various values of \( T_{SOH} \). We observe:
Figure 4. $D_{app}^{\text{max}}$ versus $B_{enc}$ for $D = 64$Bytes

Figure 5. $T_{OH}^{\text{max}}$ versus $B_{enc}$ for $D = 256$Bytes

- $T_{OH}^{\text{max}}$ increases as $B_{enc}$ increases. As the time spent for encryption decreases, the remaining time overhead increases.
- For increased application bandwidth $B_{app}$, $T_{OH}^{\text{max}}$ decreases since less time can be spent on the computer.

IV. SUMMARY

In this paper we have investigated the use of software based inline encryptors. We have integrated a number of popular encryption algorithms and measured their throughput for two computer systems configurations and different loads. To determine the suitability of software based inline encryption for multimedia applications, we have considered the multimedia application requested bandwidth, the packet size, the time overhead per packet for communication protocols processing, and the WLAN data rate.

In cooperation with AIM Engineering Inc., the Multimedia Wireless LAN Laboratory at University of Massachusetts at Amherst is developing a prototype for WLANs that support multimedia applications [2]. Our next step is to test the encryption algorithm in a multimedia testbed.

References


