QUALITY OF PROTECTION FOR MOBILE MULTIMEDIA APPLICATIONS

Chui Sian Ong, Klara Nahrstedt and Wanghong Yuan

Department of Computer Science
University of Illinois at Urbana-Champaign
{chuiong, klara, wyuan1}@cs.uiuc.edu

ABSTRACT

In traditional computer systems, security is typically provided in a one-or-nothing manner; the system is either secure or insecure. Such an approach is insufficient for pervasive environments that contain heterogenous devices with varying computing resources. The small, portable handheld devices are often left unsecured due to their limited computing power. The approach is also inadequate for multimedia applications that require security as a controllable service attribute to maintain performance quality of service to levels that are acceptable to the users. Hence, we need a tunable and differentiable security framework. In this paper, we present a Quality of Protection(QoP) framework that resolves the inadequacies of the one-or-nothing approach by providing differential security levels for different device, user and application security requirements and preferences. We show that our QoP framework is necessary for multimedia applications to achieve the best possible security and performance levels in pervasive environments.

1. INTRODUCTION

The current trends of development in pervasive computing is to continually connect smaller and more portable devices to existing wired networks via wireless technologies. These mobile devices are popularly used as thin clients to access data from remote locations or as remote control agents for application handoff in multimedia applications. For multimedia applications, most of the research is focused on smoothness of data delivery during the handoff [1]. Furthermore, due to its lower processing power, limited memory and power resources, most application developers have chosen to completely ignore security considerations on these mobile devices. This introduces incompatibilities and security flaws that open the floodgates for security attacks on the whole system.

Ideally, security should be considered as part of the application development process. However, security is a complex issue that introduces a lot of inflexibility and distracts application developers from developing the functionalities for the application. Additionally, security services can decrease the performance of an application due to resource contention. In particular, the Quality of Service(QoS) for multimedia applications will be affected and this has caused most multimedia application developers to ignore security provision completely. Hence, the goal is to provide tunable security and performance QoS services to achieve user acceptable security and performance levels.

Current solutions reuse existing security protocol or dedicated hardware security devices to provide security services. For example, at the data-link layer, security can be provided by the 802.11 network protocol. At the application layer, Secure Sockets Layer (SSL) is commonly used. Alternatively, in [2], security badges are used to provide user authentication. However, these solutions are not sufficient because the 802.11 network protocol can only be deployed on wireless devices. In addition, several security weaknesses [3] have been discovered in the 802.11 wireless network protocol. Furthermore, although SSL is considered secure, it is complicated and hard to implement and may not be feasible on handheld devices with limited processing capabilities. The hardware approach is inflexible as additional hardware is required and data encryption is not available to provide confidentiality and data integrity. More importantly, these solutions do not provide differentiable security service levels for different user and application preferences. The current solutions do not consider different amount of resources that are available on different devices. For example, a video stream handoff from a desktop to a handheld device would have different resource and security requirements before and after the handoff. Hence, existing approaches are insufficient for multimedia applications that require greater control over the overall resource usage to maintain an acceptable QoS level.

In this paper, we describe an approach that separates the development of security services from the application development, but allows application developers to seamlessly integrate tunable security services with their application. The security services can be partitioned into different security levels and are deployed to best trade off the needs of security and performance preferences. Compared to existing solutions, our approach has the advantage that it does not depend on external dedicated devices, it is also simple to deploy and it will work for both the wired and wireless networks.

In summary, this paper makes three contributions: (1) It provides a generalized Quality of Protection(QoP) framework for tuning quality of protection. The characterization of QoP with security-specific parameters provides proper construct for expressing security constraints and attributes for different application systems. (2) It describes a QoS-aware security architecture that achieves a good balance between security and performance requirements through differentiated security services. (3) It provides a QoP framework that is highly flexible and upgradable to support the latest cryptographic standards in heterogenous environments.

In section 2, we describe the QoP model. In section 3, we discuss QoP specific semantics and security service meta-data description. In section 4, we present the QoP architecture. The experimental results are presented in section 5 and we conclude the paper in section 6.

2. QOP MODEL

The QoP model is designed to be an extension for existing QoS models, where the QoP model differs from the QoS model in pro-

vision of different types of operations. QoS models, e.g DiffServ and IntServ, define operations along the end-to-end paths to check, allocate, enforce and adapt sufficient amount of resources for performance sensitive data. QoP models need to have security operations that (a) check that the access privileges to data and resources via authentication, authorization or other access control operations, (b) ensure the integrity, copyright, confidentiality of the data via encryption, watermarking and other security operations at the source and other important security points, (c) adjust security levels according to the security requirements. In this paper, our QoP model considers the user authentication operation during the setup phase, data encryption operation during the transmission phase and the overall adjustment of the application security levels.

Figure 1 presents an example of the video on demand (VoD) application extended with the QoP model. The transmission phase model captures, at the logical level, the application's functional graph, security points, and the relevant QoS and QoP meta-data.

The graph edges dictate the order in which each of these tasks are executed. The functional graph is marked with pairwise security points, S and S^{-1} along the path of data transmission. Each pair of security points carries QoP meta-data that represent a set of QoP attributes to determine the needed security services for that segment of data transmission path. Thereafter, the security points act as decision engines that decide the most suitable QoP level of security to be executed and thus forward them to the appropriate security operation tasks. For example, the data from the video server must be encrypted before it is transmitted over the internet. Then, before the data can be processed by the transcoder, the data must be decrypted with the appropriate QoP parameters first. If no security operations are required, the data is transmitted using the alternate dotted path.

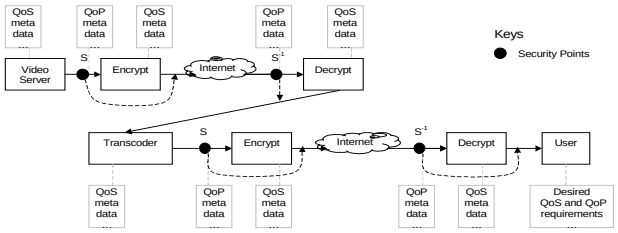

Fig. 1. QoP model (encryption) during transmission phase.

The user and application QoS and QoP requirements are specified in terms of QoS and QoP parameters prior to the invocation of the applications. We present a brief discussion of the QoP meta-data that are considered in the framework in the next section.

3. QOP META DATA

The semantics for each security service is expressed through QoP meta-data. Hence, in this section, we will discuss in detail the QoP parameters and QoP reward profiles that are part of the QoP meta-data description. This section also shows that although QoP appears similar to QoS, there are many underlying differences and complexities due to specific QoP service requirements that are not considered in existing QoS models. Moreover, our QoP model allows us to clearly observe the interaction between QoP and QoS and this justifies the need to explore QoP as an extentsion to QoS.

3.1. QoP Security Services

Our QoP model introduces, at this point, two main security services; user authentication and data encryption. The authentication service resides in the setup phase of a multimedia service, it has the nature of an event, hence the QoS of this security service is to execute the service in a bounded amount of time to minimize the overall setup time. The encryption service is a multiple time event. It is applied at points where secure multimedia processing must happen. It is different from a multimedia decoding service as it is applied only at security points of the media stream, not at every node of the end-to-end multimedia path. Furthermore, depending on where the security points are, the security results and goals are different. For example, the first time encryption is applied at the first security point, the data is transformed from the $unsecured$ plaintext state to a $secured$ ciphertext state. However, when encryption is subsequently applied again at intermediate security points, we require that the security levels are maintained in an end-to-end manner. For instance, after an intermediate transcoding task, at that security point, a different encryption function may be applied on the transcoded data to give us a different ciphertext data. The requirement is that the security level on the transcoded data must be the same as the security level on the data before the transcoding process. Finally, at the last security point, the secured ciphertext data is decrypted and we have the desired plaintext data.

3.2. QoP Parameters

QoP parameters represent part of the security service semantics and are used to quantify different security attributes. The QoP parameters is dependent on security goals such as confidentiality, integrity and authentication. Although these security goals cannot be easily measured [4], however, it is still possible to quantify security by expressing QoP parameters in terms of cryptographic variables such as encryption algorithm and encryption key lengths. In general, a longer key provides a higher level of security.

The syntax and semantics for QoP parameter specification is dependent upon the security scheme selected. For example, for DES encryption, the tuple $\langle encryption\ algorithm, encryption\ key\ length, encryption\ blocksize \rangle$ is invalid syntatically as the DES operates on fixed block size. On the other hand, the same tuple is syntatically valid but may or may not be semantically valid for the AES encryption algorithm. The block size for the AES encryption must be specified in multiples of 8bytes.

Time can also be a very valuable QoP parameter. For instance, in broadcast news, an important QoP parameter is the time interval to keep the data secure. This means that the information will require no security services once the news has been publicly broadcast. Therefore, the QoP meta-data for encryption service can be further specified in the following form $\langle\ content\ type, interval\ of\ security, encryption\ algorithm, encryption\ key\ length, encryption\ blocksize \rangle$.

In general, the tunable QoP parameters are multi-dimensional attributes and can be specified as a vector.

$$\begin{bmatrix} & A_L & E_L & W_L & A_M & E_M & W_M & A_H & E_H & W_H \\ V & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ Al & P & - & - & MD & DES & - & S & AES & - \\ K & - & - & - & - & 64 & - & 512 & 256 & - \\ I & - & - & - & - & - & - & - & - & - \end{bmatrix}$$

The first row represents the available security operations such as authentication(A), encryption(E) and watermarking(W) for each level (low, medium and high) of QoP. Clearly the vector can be easily extended for other security operations. The second row con-

tains the binary value that decide if each operation is valid(V)or not. The third, fourth and fifth row are the tunable security operation specifics parameters such as key lengths(K), cryptographic or authentication algorithms(Al) and security time interval(I). Some of the selectable authentication algorithms include password(P), messageDigest(D), signature(S) authentication methods. Additional rows can be added for considering other QoP parameters.

3.3. QoP Reward Profiles

The QoP reward profiles are the key for providing differential levels of security. The reward profile can be considered a multidimensional function indicating the security benefit the application user receives as a function of the QoP parameters provided. It is a mapping relation from the QoP parameters to its security strength. The reward profile will be used by the resource management services to produce a set security levels achievable by the system. This set of values is then presented to the user who uses it to choose his security preferences and requirements. Therefore, unlike the one-or-nothing approach, instead of providing no security at all when the resources is limited, it is now possible to provide security at a lower level yet still satisfying the user's security requirements.

Table 1. Example Security Profile for MPEG Video Encryption.

Algorithm	Key Length (bits)	Security Strength	User Specific Security Levels
DES	64	1	low
AES	128	2	low
RC6	128	3	low
DES	192	4	medium
AES	192	5	medium
AES	256	6	high

Table 1 presents an example of a reward profile for encryption on video data. A larger value indicates a stronger security level. The example only considers two QoP parameters, a real reward profile will have more QoP parameters and more levels of security strength. The resource management services, based on available resources, determine which of the levels from the reward profile can be achieved and generate a relation in the following manner: $\langle DES, 64 \rangle < \langle AES, 192 \rangle < \langle AES, 256 \rangle$. This is then further translated correspondingly to user-specific terms such as low, $medium$, $high$ security so that the user can easily select the desired level of security.

4. QOP ARCHITECTURE

The QoP architecture is implemented at the middleware layer. In this section, we present the middleware components that assist the application to provide tunable security services and to enforce QoP. These components allow the user to fill in specific QoP requirements for the QoP application model that they wish to invoke. These components are responsible for attaching the proper security attributes at each security point according to the selected QoP requirements. Furthermore, we present a QoS-aware security architecture that shows the overall interaction of QoP and QoS provision. Figure 2 shows the various components and how they interact within the QoP architecture.

1. The *Application QoS Specifier* component obtains the desired QoS parameters from the application. For example, for a video on demand application, the QoS parameters may be frame rate or jitter or both.

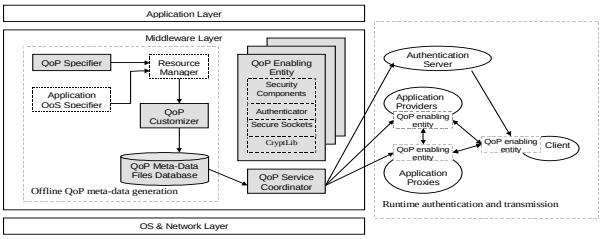

Fig. 2. System Architecture and component protocol interaction.

2. The *QoP Specifier* determines the various QoP meta-data to be considered. For example, security algorithm, key lengths, chaining or non-chaining encryption, number of rounds of encryption and the encryption block size.
3. The *resource manager* obtains the specified QoP and QoS parameters and matches them against their respective reward profile to determine the set of values that will be feasible given the available resources.
4. The *QoP customizer* may contain additional application specific rules and limitations, such as requiring QoS to have a higher priority over QoP. In this way, when the QoP customizer obtains the feasible set of security levels from the resource manager, it will further refine the executable levels according to these application specific rules and outputs the security levels to a QoP meta files that will be stored in a centralized database.
5. The *QoP Service Coordinator* extracts the security requirements from the QoP meta file and relays the information to the various security points within each application by filling in the required security attributes. This ensures that each component executes the required security protocols correctly during application runtime.
6. The *QoP-enabling entity* is composite component that provides the core set of QoP services such as authentication and encryption to the applications. *CryptLib* is a cryptographic library that we built to provide key generation, encryption and decryption functions. QoP service components such as the *Authenticator* and *Secure Sockets* are built on top of CryptLib. The *Authenticator* is responsible for authentication services. The *SecureSockets* simulates security points and is responsible for data encryption services specified in its security attributes. Communications between the QoP-enabling entities are governed by security protocols. Hence, the system can be easily upgraded with the latest cryptographic standards by changing $CryptLib$ only.

As shown in Figure 2, QoP can be easily provided by including the QoP-enabling entity as a middleware component in the applications. In this way, the middleware components (QoP service coordinator and QoP-enabling entities) manage all the QoP communications without interfering with other parts of the application. The QoP meta-data file is generated and stored in the meta-data files in XML format during the offline phase just before the application instantiation. This ensures that the QoP meta-data file accurately reflects the amount of resources that is available at that point of time. The available levels of security are presented to the user during the authentication phase. After the user is correctly authenticated, the QoP Service Coordinator distributes the user's

selected QoP levels to the various security points. Hence, at runtime, the application is properly integrated with both QoS and QoP requirements and is able to achieve the best performance and security tradeoffs.

5. EXPERIMENTAL RESULTS

We validate our QoP framework on the VOD application. Figure 3 shows the experiment setup. The considered QoP parameters are authentication method, encryption algorithm and key lengths.

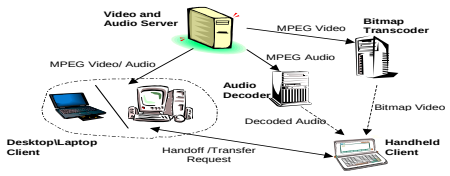

Fig. 3. Experiment Setup.

The desktop and laptop are connected to the video and audio server with 100Mbps wired Ethernet. The handheld device is connected to the Bitmap Transcoder and Audio Decoder with 11Mbps 802.11b wireless network. The mobile user can handoff the media stream from the handheld to the desktop or laptop and vice versa. The QoP meta-data files database contains one QoP meta-data file to represent the security requirements for the application. However, each of the handheld, desktop or laptop may choose a different level of QoP. A desktop (PIII 733MHz), a laptop (PII 266MHz) and a Handheld Jornada (Hitachi SH-3 150MHZ) were used for the experiment.

In the experiment, each user must be authenticated before the media playback starts. Depending on the chosen security level, (low, medium or high), data encryption is performed before and after each multimedia processing function, i.e. before and after the audio decoder. Hence each multimedia processing function also represents the security points in our experiment. Some metrics for evaluating QoP includes (a)security processing overheads and delays, (b)how secure the system is and (c)how vulnerable the system is to security attacks. In this paper,due to space limitation, we only show the security processing overheads. We measured the delays incurred by each of security operations with the QoP parameters varied according to the QoP meta-data file. The results are presented in Table 2 and Table 3. Each table lists the authentication method and encryption algorithm from the least secure to the most secure.

Table 2. Authentication Delays.

Authentication Method	Delay on Jornada (ms)	Delay on PII (ms)	Delay on PIII (ms)
Password	284.65	189.80	29.52
MessageDigest	313.75	189.81	28.14
Signature (512bits)	3969.58	1558.53	463.62
Signature (1024bits)	59077.60	15655.50	6377.92

Table 3 presents the delay values to encrypt each data packet. Each audio packet is 4608 bytes long and each bitmap packet is 5120 bytes long. The encryption delays on the desktop and laptop was less than 1ms, hence considered negligible. During user handoff, mutual authentication was performed and the extra delay incurred was measured to be 2ms.

Table 3. Encryption Delays on HP Jornada.

Algorithm (Key length)	Audio Delay (ms)	Video Delay (ms)
DES (64bits)	16.24	15.5
Triple DES (196bits)	30.95	41.2
AES (256)	16.95	15.3

The QoP-aware media player was as easy to use as the original player application. Most of the security services were performed by the underlying middleware components unknowingly to the user. The authentication process was fast except for signature authentication with 1024bits RSA key generation on the Jornada. However, this problem can be easily circumvented during the QoP customization process, where the developer can easily specify a rule for the *QoP-customizer* to disallow generation of 1024bits RSA keys for user authentication on the Jornada when performance quality has higher priority over security requirements.

Furthermore, the video and audio playback quality was mostly unaffected by the mutual authentication or encryption and decryption process. The mutual authentication delays were much less than the user handoff overhead and was thus not perceptible to the user. The only instance, when audio quality was significantly degraded, was when Triple-DES was used. Long pauses were observed during the playback. The jitters were caused by untimely audio data packet arrival which is a direct consequence of the additional delays introduced by data encryption. In this case, the QoP-customizer could be used to restrict the use of Triple-DES encryption and use AES instead.

6. CONCLUSION

We have presented a generalized QoP framework to provide application security in a differentiable and quality-aware manner. Our experiments have provided favorable results on the applicability, flexibility and usefulness of the QoP system.

7. ACKNOWLEDGEMENT

This work is supported by NASA grant, NASA NAG 2-1406, Office of Naval Research MURI grant, NAVY CU 37515-6281, and NSF CISE Infrastructure grant, NSF EIA 99-72884.

8. REFERENCES

[1] R. Karrer and T. Gross, "Dynamic Handoff of Multimedia Streams," in *Proceedings of International Workshop on Network and Operating System Support for Digital Audio and Video*, 2001.

[2] Bo Zou, "Mobile ID Protocol: A Badge-activated Application Level Handoff of a Multimedia Streaming To Support User Mobility," M.S. thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, 2000.

[3] P. Bahl, A. Balachandran, and S. Venkatachary, "Secure Broadband Wireless Internet Access in Public Places," in *Proceedings of IEEE International Conference on Communications*, June 2001.

[4] Stefan Lindskog and Erland Jonsson, "Adding Security to Quality of Service Architectures," in *Proceedings of the SSGRR Conference*, Aug. 2002.