Implementing a VIPSec Based Application for Handhelds: Design and Optimization Issues

Spyros Kopsidas
University of Thessaly, Computer Engineering & Communications Dpt.
37 Gklavani str.
38221, Volos, Greece
+302421074553
spyros@uth.gr

Dimitris Zisiadis
University of Thessaly, Computer Engineering & Communications Dpt.
37 Gklavani str.
38221, Volos, Greece
+302421074977
dimitris@uth.gr

Leandros Tassiulas
University of Thessaly, Computer Engineering & Communications Dpt.
37 Gklavani str.
38221, Volos, Greece
+302421074980
leandros@uth.gr

ABSTRACT
Voice over IP technology enabled the growth of Internet telephony applications that make use of public common infrastructures to provide voice and data communications to their users. The growth of wireless Internet enables users to connect to the network from different locations using mobile Internet devices. Security is a major concern for mobile Internet telephony users, yet the lack of secure inline key exchange mechanisms is one of the major drawbacks for the use of these applications for business use. The intrinsic vulnerabilities of wireless networks make the VoWiFI case even worse while compromising security of a call is as easy as breaching human administration security of the service network. In this paper we describe the design and implementation of a secure VoIP application for handheld devices that is using the strong security mechanisms of Voice Interactive Personalized Security protocol (VIPSec). We analyze the architectural fundamentals and we present the implementation elements of the application. User sign-in and other subsidiary procedures follow the Client-Server model while the media path is direct between the users. The application is developed in Java and is suitable for mobile devices running Windows CE or Linux, essentially securing end-to-end voice, video and data communications in wireless communications.

Keywords
VoIP applications, mobile devices, personal communications, privacy

1. INTRODUCTION
Applications for voice communications over the Internet, using VoIP technology, are mainly targeting desktop computers. Skype [1], VoIPBuster [2] and MSN messenger [3] are the dominating applications in this area. They are implementing the Client-Server model for user authentication and secondary processes regarding user calls while the media path is usually direct between the peers, without any involvement of the service network; Skype "super node" communications is an exception in this case.

Voice and data packets travel through multiple public Internet infrastructures like Ethernets, WiFi hotspots or wireless ad hoc networks. There are major security concerns, especially in wireless environments, which have been studied thoroughly [7][8][9][10][19]. Skype successfully addresses typical eavesdropping attacks, yet none of the above VoIP systems overcomes sophisticated Man-In-The-Middle (MITM) attacks. Even though the above systems are well accepted by the Internet community and broadly used in the home/office environments, yet their service offering is not equally strong in the mobile case, where lightweight and targeted applications are preferred.

In this paper we provide the design and implementation analysis of a prototype application for mobile devices which embeds VIPSec protocol's features for secure inline key exchange with first level verification mechanism (vocal confirmation). Security in VIPSec is achieved through dynamic cooperation of the peers with no prior arrangements and requirements (i.e. out of band exchanged keys, shared secrets etc). Simplicity, ease of use, user friendliness and effectiveness along with low demanding requirements on the mobile devices are the goals achieved by our proposal. We have implemented the VIPSec application in Java [22]. Java is a programming language originally developed by Sun Microsystems. Java applications are typically compiled to bytecode, although compilation to native machine code is also possible. At runtime, bytecode is usually either interpreted or compiled to native code for execution, although direct hardware execution of bytecode by a Java processor is also possible. Java is ideal for handheld applications development, since it is supported by most of the various mobile device types. These features are in line with our objective to provide ubiquitous access to the application with minimal restrictions to the client machine platform.

The VIPSec protocol has been thoroughly analyzed in [4]. This paper focuses mainly on the design and implementation of the application as well as optimization details for handhelds. The content of the paper is organized as follows: related work is provided in Section 2; Section 3 deals with system design, presenting the application's architectural model; Section 4 addresses system implementation issues, taking into consideration the necessary optimization due to the nature of the mobile devices; performance of the mobile client application is examined

Mobimedia'07, Month 8, 2007, Nafpaktos, Aitolokarnania, Greece.

in Section 5, providing measurements on delay overheads for a typical handheld. Finally, conclusions and potential directions for future work are provided in Section 6.

2. RELATED WORK

As mentioned before, there is no secure VoIP application developed specifically for Internet connected mobile devices as the majority of VoIP systems have been designed for desktop use. Besides their broad public use they suffer from weaknesses, originating from their network architecture design or their lack of encryption methods and algorithms in the source code. Secure key exchange and data integrity checks are not guaranteed in current implementations of VoIP applications.

Both MSN messenger and VoIPBuster use unsecured media paths, exposing them to eavesdropping and MITM attacks [18]. Skype on the other hand, supports a higher level of security against eavesdropping attacks but still does not cope with more sophisticated MITM attacks. Skype's security model is considered to be a black-box, since the Skype Company has not published its architecture details. Nevertheless, P. Biondi and F. Desclaux have lately published a complete Skype analysis, from which it is easily concluded that a MITM who controls a router involved in a Skype connection is theoretically able to unleash a successful attack modifying the exchanged keys inline [20].

A new security protocol for voice communications called ZRTP [5][11] and a corresponding beta application called ZFone [6] have been recently introduced by PGP's father Phil Zimmerman. Zfone achieves superior security in comparison to the above systems, based on the verification through the human voice of a small authentication string. In addition, Zfone provides a second layer of authentication against a MITM attack, based on a form of key continuity. It does this by caching some hashed key material for subsequent calls, to be mixed in with the call's Diffie-Hellman shared secret, giving it key continuity properties. If the MITM is not present in the first call, he is locked out of subsequent calls. Although a valuable security add-on, the basic weakness of this method is that it is device dependent, since the users have to use the same equipment each time in order to call each other. Other approaches, similar to Zfone although less effective have also been published [12] [13], yet there is no implementation based upon them.

VIPSec follows a different key exchange mechanism. The communicating parties exchange random numbers (stronger option: biometric type objects) encrypted with their session private keys at the start of the call, followed by the exchange of their session public keys. In the next step, the caller produces a symmetric key, encrypts it with the callee's public key and sends it to him. Finally, a symmetrically encrypted communication channel is established and the two users vocally confirm the exchanged objects (stronger option: video confirmation). In VIPSec the security relies only on the effectiveness of the algorithms and the lengths of the keys used, avoiding the hash production phase which may constitute a security weakness.

Both ZRTP and VIPSec are vulnerable to voice-mimic attacks although that kind of attacks is not considered to be yet practical. The stronger verification procedure of VIPSec (confirmation using video) makes it more difficult for an attacker to achieve a successful user impersonation though.

3. SYSTEM DESIGN

The implemented VIPSec prototype application provides reliable privacy and security, as well as high flexibility in terms of network communications. The architecture of the application applies the principles of the client-server model regarding the control data and the client-client model regarding the media path. Well-known multimedia development techniques have been adopted in order to provide structural and technological compatibility. For enhancing system's usability secure directory services have been designed and implemented. Since the sensitive nature of the application (real-time) demands an affordable quality of service, the use of standard encryption and voice encoding functions and algorithms was the only acceptable option.

Specifically, the key administrative element in our architecture is the central directory server, which provides user registration and sign-in as well as directory list and lookup functions. Upon registration, users provide all relevant personal information like name, e-mail etc in order to register to the directory server. The directory server makes online users list available to all signed-in users as well as user lookup function using the e-mail address or nickname as the search key. The server associates the user identification elements (e-mail, nickname) with its current IP address upon sign-in. The server returns location information to the initiating client of a call. The client in turn initiates a media path directly to the other client over the Internet without any server intervention. The VIPSec handshaking process is then followed, using random numbers as the exchanged objects; after successful vocal verification of each other's session numbers the users are able to communicate securely. Fig. 1 below demonstrates the communications diagram. In the first step, Alice and Bob sign into the directory server. Alice queries the server for Bob's IP address and calls Bob directly. Next, a VIPSec handshaking procedure takes place in order to secure the communication channel [14]. Finally, Alice and Bob confirm the exchanged numbers. If the confirmation is successful the exchanged symmetric key secures the media path. Otherwise communication channel is considered to be compromised [15].

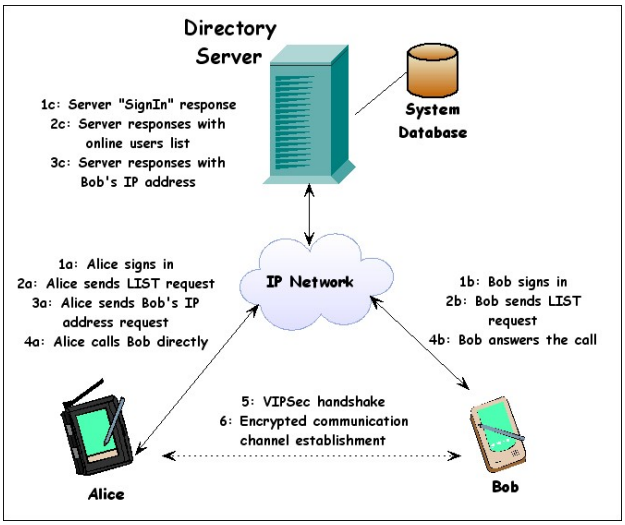

Figure 1. Communications diagram

The client application architecture is depicted in fig. 2. As shown, the user interacts with the application through the microphone and speakers of the device as well as through the relevant graphical user interface (GUI). The user through the GUI performs all the necessary actions related to the call (call origination, answer, etc.).The complete server and client GUIs are thoroughly described in the next section. A voice codec is included in order to enable packet voice communication. The Symmetric Encryption mechanism is responsible for the encryption/decryption of the voice packet stream that goes to or comes from the network, respectively. A VIPSec mechanism, which ensures connection security and user privacy is provided. Internally, the VIPSec mechanism implements functions for asymmetric and symmetric cryptography in order to achieve key exchange and secure channel establishment. The asymmetric cryptography function provides 2048 to 3072 bits asymmetric key generation and encryption/decryption of the exchanged objects. It also generates the 1024 bits asymmetric keys needed for the client-server communication link (simple Diffie-Hellman protocol). A 128 or 160 bits AES symmetric key is agreed during the VIPSec handshaking, which in turn is provided to the Symmetric Encryption mechanism mentioned above and is used to encrypt the specific session. Finally, as the server database we use an open-source of My-SQL.

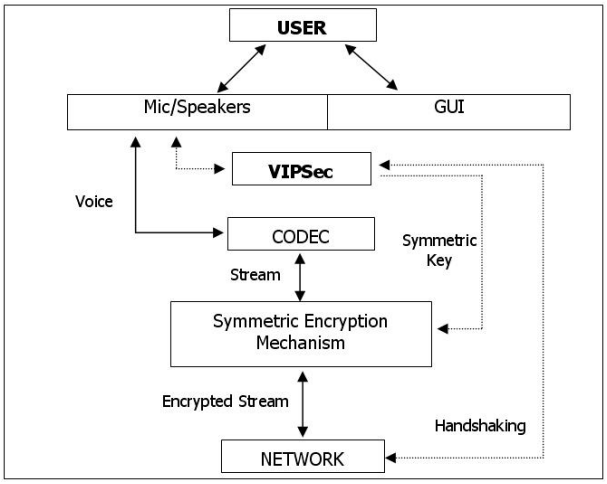

Figure 2. Architecture of the mobile VIPSec based client application

4. SYSTEM IMPLEMENTATION

The processing components of the implemented system and the core multimedia elements are developed using the Java Media Framework (JMF) [23]. JMF enables video, audio and other time-based media to be added to Java application or applets. This optional package, which can capture, playback, stream and transcode multiple media formats, gives multimedia developers a powerful toolkit to develop scalable, cross-platform technology. Based on the functions provided by JMF, plus the Speex [21] audio codec, the Diffie-Hellman [16] asymmetric cryptography algorithm and the Advanced Encryption Standard (AES) [17] symmetric cryptography function, which are provided by standard open-source libraries, our system achieves end-to-end security functionality.

AES, also known as Rijndael, is a fast and reliable block cipher adopted by the majority of the industry and academic organizations. It has been analyzed extensively and is now broadly used as was the case with its predecessor, the Data Encryption Standard (DES) [24]. AES is one of the most popular algorithms used in encrypting communications symmetrically and performs well when executing on 32 bit RISC CPUs.

The length of the cryptography keys is an option selected by the user. In our implementation 2048 to 3072 bits and 128 to 160 bits keys are available for the asymmetric and the symmetric cryptography mechanisms, respectively. Regarding the transport of the exchanged data, we chose to use TCP client-server connections (control data), TCP client-client connections for the VIPSec handshaking process and UDP client-client connections for voice data (media path). As already mentioned, our system consists of the server and the client applications. The functions provided by each application's GUI are described in tables 1 and 2 below.

Table 1: Server GUI functions

Id.	Function
1.	Server start
2.	Server stop
3.	User management (add, delete, edit)
4.	Connections monitoring
5.	Log file

Table 2: Client GUI functions

Id.	Function
1.	User registration
2.	Sign-in
3.	Logout
4.	List online users
5.	User lookup
6.	Make call
7.	Answer call
8.	Call termination
9.	Log file

The server application is responsible for connection management and general call control. The client signs-in the server following the specified procedure. With the connection establishment, the client automatically sends a "LIST" request to server in order to receive a list of the online users. When a user calls another, a call initialization process takes place. The first client (which makes the call) requests the second's client (which receives the call) IP address from the server and normally the server responses with the IP address. When the first client receives it, uses it to make a direct connection to the second client. In addition, the status of the two clients in the database is transparently set to "Busy". That means that no other call can be initialized to the two users until their status returns to "Online". The status is modified again when the call process terminates.

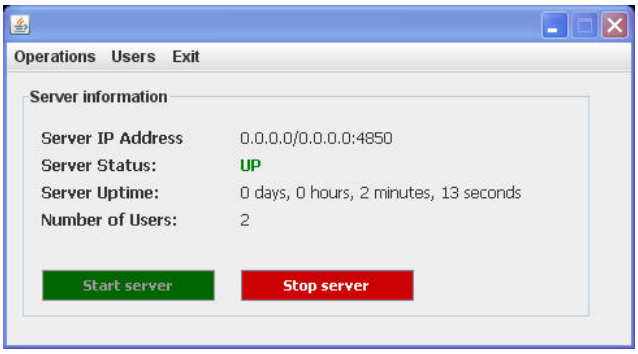

Figure 3. The server administrator GUI

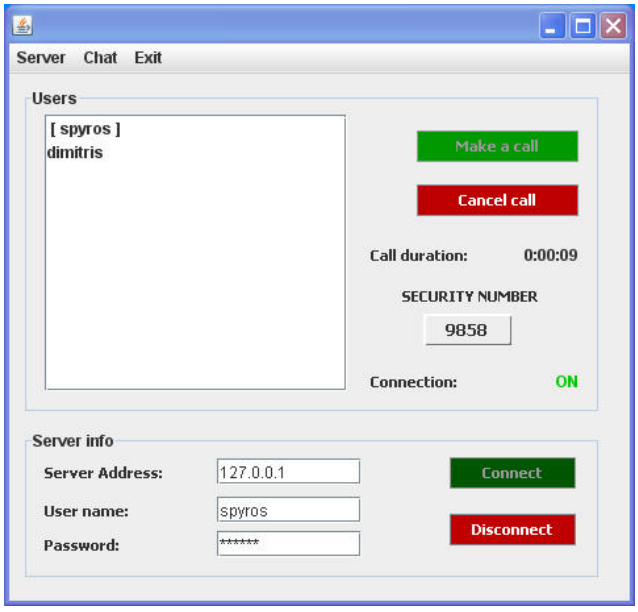

Figure 4. The mobile client application GUI

All generated runtime and communication messages are thoroughly logged both on the server side as well as on the client side. Snapshots of these files are shown in fig. 5 and 6. In our prototype even the generated keys are logged mostly for debugging reasons.

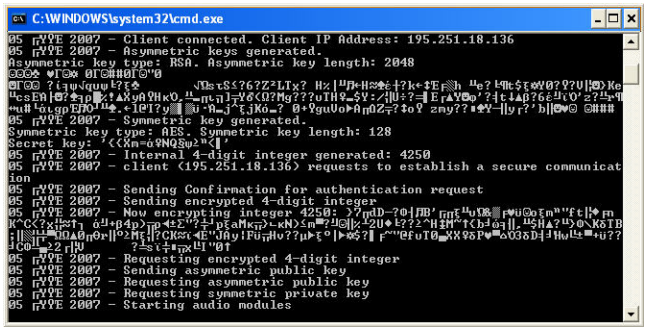

Figure 5. The server log file

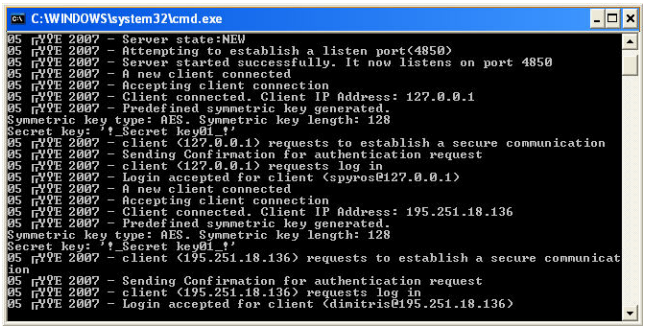

Figure 6. The mobile client log file

Regarding the runtime phase certain performance issues were taken under consideration and some custom optimization methods, described below, were adopted.

Asymmetric key pre-generation is used to enhance the application's performance. Asymmetric key pairs are generated at times the mobile device's CPU is idle and there are no power constraints (i.e. the device is charging or it is running on batteries with more than 30% battery time left) and are consequently stored in the database. Call setup time is minimized when the above functions are enabled. Using the pre-generated asymmetric key database the handshaking procedure is faster, by skipping to the next step, making the application more efficient.

A repository for session symmetric keys that were securely established in prior sessions is also implemented. These keys can be re-used in case the same users wish to communicate again in the near future, resulting in significant savings on both user time and device resource utilization. The key re-use window is kept below 24 hours for security reasons.

5. EXPERIMENTAL RESULTS

For the calculation of the overheads imposed by the protocol primitives during its operation we used a typical user handheld PC (PDA) equipped with a 400MHz ARM9 RISC CPU and 64MBs of RAM. We measured VIPSec's overheads for the following primitive protocol's operations:

I. Time to produce asymmetric cryptography keys. This can be performed during the connection establishment phase or in advance.

II. Time to produce symmetric cryptography key. This can also be performed during the connection establishment phase or in advance.

III. Time and CPU utilization to simultaneously encrypt and decrypt voice data packets. This is the normal communication phase where signatures have been verified and secure voice traffic is exchanged between the two parties.

VoIP applications are delay and jitter sensitive. Jitter is not affected at all by the application though as it relies entirely upon network transport and QoS parameters. The mobile client application adds only a small fraction of time (tiny lag) at the transmission of the 1st packet in a row, which is negligible as expected, something that was proven in the experiments.

We used Speex to encode a wav stream with 8 KHz sampling rate and 16bits sample size (128Kbps rate) into an 8 KHz and 16Kbps encoded voice stream which is acceptable for VoIP communications. The voice payload in each packet is 40 bytes. We performed multiple runs and we calculated the mean times and CPU utilization percentages of the runs. Deviation was negligible to show. Tables 3 and 4 show the respective measurements for primitive operations I and II (asymmetric and symmetric key generation respectively).

Table 3: Time to generate asymmetric cryptography keys

Algorithm	Key size (bits)	Time (msec)
DH	1024	198.212
	2048	845.621
	3072	1274.429

Table 4: Time to generate symmetric cryptography keys

Algorithm	Key size (bits)	Time (msec)
AES	128	11.341
	160	12.543

For primitive operation III, four metrics are used for evaluation of the system's performance:

1. Metric M1: time to encode the voice payload of a packet (40 bytes) with Speex.
2. Metric M2: time to decode the voice payload of a packet with Speex.
3. Metric M3: time to encrypt the voice payload of an outgoing packet and simultaneously decrypt the voice payload of an incoming encrypted packet.
4. Metric M4: CPU utilization when performing all of the above tasks simultaneously.

M1 and M2 are fairly simple to calculate. They are based only on codec's characteristics and are independent of the encryption algorithm. Table 5 shows the relative values for a 40 byte packet voice payload.

Table 5: Time to encode/decode a voice packet

Packet Voice payload	Metric M1	Metric M2
	Packet encode time	Packet decode time
40 bytes	1.423 msec	1.528 msec

Voice encoding and decoding are functions that are performed simultaneously during any VoIP session. This is noted as metric M3 in our case and is of great interest in evaluating our proposal. M3 effectively is the additive delay due to the VIPSec encryption and decryption mechanisms featured by the application. Table 6 below provides the relevant values for M3 as well as M4.

Table 6: Time and processing overheads during encrypted VoIP sessions

Algorithm	Key size *(bits)*	Metric M3	Metric M4
		Time to Encrypt & decrypt voice payload of a packet (40 bytes) *(msec)*	CPU Util. (%)
AES	128	0.114	27
	160	0.125	31

The voice channel delay budget is not affected significantly by the protocol operation. As it is shown by the experiments (metrics m1 to m4) the one way delay sums up close to 1.5msec.

The time to produce asymmetric cryptography keys is fairly small, well below 1 sec (846 msec is the maximum key generation time for a 2048 bits DH key) whereas the time to produce symmetric keys is less than 1 msec. As stated in Section 4, even if asymmetric keys generation times can be significantly larger for highly loaded CPUs, the pre-generated asymmetric keys database addresses effectively with this issue.

The experimental results show that runtime overheads of the proposed architecture are minimal and the implementation analysis is both functional and effective.

6. SUMMARY

Architecture and implementation concepts for the development of a prototype application for handheld devices are addressed, making use of VIPSec protocol's secure channel establishment primitives. The design goals are stated and evaluated accordingly. The implemented application achieves a high degree of privacy for Internet telephony VoIP sessions. Dynamic user interactivity is also supported due to the nature of the VIPSec protocol.

Effectiveness and completeness of the Java implementation according to the defined system design concepts is shown. Moreover, optimization analysis for maximizing application performance at run-time level is provided.

Future directions include support for 2nd level VIPSec security procedure (through video confirmation), enhancing scalability of the application through the incorporation of adjustable transmission rate and quality features in the application design by supporting more codecs and finally further improvement of application performance through support of elliptic curve cryptography. .

7. REFERENCES

[1] http://www.skype.com

[2] http://www.voipbuster.com

[3] http://www.msn.com

[4] Kopsidas Spyros, Zisiadis Dimitris and Tassiulas Leandros, "*Voice interactive personalized security (VoIPSec) protocol: fortify internet telephony by providing end-to-end security through inbound key exchange and biometric verification*",

Hot Topics in Web Systems and Technologies, 2006. HOTWEB '06. 1st IEEE Workshop on, Vol., Iss., Nov.2006, pp.1-10

[5] Phil Zimmerman, "ZRTP: Extensions to RTP for Diffie-Hellman Key Agreement for SRTP", Internet Draft

[6] http://zfoneproject.com

[7] http://en.wikipedia.org/wiki/Evil_twin_(wireless_networks)

[8] Savola R., Lehtonen S. and Röning, J., "*Information Security Threats in Popular and Emerging Wireless Technologies*", Proceedings of the 6th Annual Security Conference. Las Vegas, NV, USA, 11 - 12 April 2007. The Information Institute. Washington, DC, USA (2007), 25-1 - 25-11

[9] Feng Cao and Malik S., "*Security analysis and solutions for deploying IP telephony in the critical infrastructure*", Security and Privacy for Emerging Areas in Communication Networks, 2005. Workshop of the 1st International Conference on, Vol., Iss., 5-9 Sept. 2005 pp. 171- 180

[10] G. Me and D. Verdone, "*An overview of some techniques to exploit VoIP over WLAN*", Digital Telecommunications, 2006. ICDT '06. International Conference on, Vol., Iss., 2006 p. 67

[11] Phil Zimmerman, Jon Callas, *"ZRTP and ZFone"*, NOMS 2006, 3rd April 2006.

[12] Michael T. Goodrich, Michael Sirivianos, John Solis, Gene Tsudik and Ersin Uzun, "*Loud and clear: human verifiable authentication based on audio*", Distributed Computing Systems, 2006. ICDCS 2006. 26th IEEE International Conference on, Vol., Iss., 2006 p. 10

[13] Cagalj M., Capkun S. and Hubaux J.-P., "*Key agreement in peer-to-peer wireless networks*", Proceedings of the IEEE, Vol.94, Iss.2, Feb. 2006 pp. 467- 478

[14] http://vipsec.inf.uth.gr/PresentApplet1.html

[15] http://vipsec.inf.uth.gr/PresentApplet2.html

[16] http://en.wikipedia.org/wiki/Diffie-Hellman

[17] http://en.wikipedia.org/wiki/Advanced_Encryption_Standard

[18] King Tom, "*Packet Sniffing In a Switched Environment*", SANS Institute, July 2006, http://www.sans.org/reading_room/whitepapers/networkdevs/244.php

[19] Deckerd Gary, "W*ireless attacks from an intrusion detection perspective"*, SANS Institute, November 2006, http://www.sans.org/reading_room/whitepapers/honors/1681.php

[20] Philippe Biondi, Fabrice Desclaux, *"Silver needle in the Skype"*, BlackHat Europe, March 2nd and 3rd 2006.

[21] http://www.speex.org

[22] http://en.wikipedia.org/wiki/Java_(programming_language)

[23] http://java.sun.com/products/java-media/jmf/

[24] http://en.wikipedia.org/wiki/Data_Encryption_Standard