UNIVERSITAS INDONESIA **UNIVERSITÉ DE BRETAGNE-SUD**

PORTING IN WINDOWS AND CRYPTOGRAPHY SERVICE FOR DoDWAN

TESIS

ANDY TRIWINARKO
0906644114

FAKULTAS TEKNIK
PROGRAM STUDI TEKNIK ELEKTRO
DEPOK
JUNI 2011

UNIVERSITY OF INDONESIA

UNIVERSITY OF SOUTH-BRITTANY

PORTING IN WINDOWS AND CRYPTOGRAPHY SERVICE FOR DoDWAN

TESIS

Diajukan sebagai salah satu syarat untuk memperoleh gelar Master

ANDY TRIWINARKO
0906644114

FAKULTAS TEKNIK
PROGRAM STUDI TEKNIK ELEKTRO
DEPOK
JUNI 2011

HALAMAN PERNYATAAN ORISINALITAS

Tesis ini adalah hasil karya saya sendiri,

dan semua sumber baik yang dikutip maupun yang dirujuk

telah saya nyatakan dengan benar.

Nama	:	Andy Triwinarko
NPM	:	0906644114
Tanda tangan	:	
Tanggal	:	4 Juli 2011

UCAPAN TERIMA KASIH

Puji syukur saya panjatkan kepada Tuhan Yang Maha Esa, karena atas berkat dan rahmat-Nya, saya dapat menyelesaikan tesis ini. Penulisan tesis ini dilakukan dalam rangka memenuhi salah satu syarat untuk mencapai gelar Magister Teknik Jurusan Teknik Elektro pada Fakultas Teknik Universitas Indonesia. Saya menyadari bahwa, tanpa bantuan dan bimbingan dari berbagai pihak, dari masa perkuliahan sampai pada penyusunan tesis ini, sangatlah sulit bagi saya untuk menyelesaikan tesis ini. Oleh karena itu, saya mengucapkan terima kasih kepada:

1. Pascale Launay dan Frédéric Guidec selaku dosen pembimbing yang telah menyediakan waktu, tenaga, dan pikiran untuk mengarahkan saya dalam penyusunan tesis ini;
2. Seluruh personil Group CASA, Laboratorium VALORIA yang telah banyak memberikan semangat dan saran, sehingga saya dapat berintegrasi dengan baik di lingkungan laboratorium.
3. Istri, anak, orang tua dan keluarga saya yang telah memberikan bantuan dukungan material dan moral; dan
4. teman-teman di UBS dan UI yang telah banyak membantu saya dalam menyelesaikan tesis ini.

Akhir kata, saya berharap Tuhan Yang Maha Esa berkenan membalas segala kebaikan semua pihak yang telah membantu. Semoga tesis ini membawa manfaat bagi pengembangan ilmu.

Depok, 4 Juli 2011

Penulis

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai civitas akademika Universitas Indonesia, saya yang bertanda tangan di bawah ini :

Nama	:	Andy Triwinarko
NPM	:	0906644114
Program studi	:	Teknik Telekomunikasi
Departemen	:	Teknik Elektro
Jenis karya	:	Tesis

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Unibersitas Indonesia hak bebas royalti non eksklusif (non-exclusive royalty-free right) atas karya saya yang berjudul: **PORTING IN WINDOWS AND CRYPTOGRAPHY SERVICE FOR DoDWAN**.

Dengan hak bebas royalti non-eksklusive ini Universitas Indonesia berhak menyimpan, mengalihmedia/formatkan. Mengolah dalam bentuk pangkalan data (database), merawat dan mempublikasikan tugas akhir saya tanpa meminta izin dari selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik hak cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di	:	Depok
Pada tanggal	:	4 Juli 2011

Yang menyatakan

Andy Triwinarko

ABSTRACT

DoDWAN (Document Dissemination in Wireless Ad hoc Networks) is the middle ware platform created by CASA group of the VALORIA laboratory, that support the model of communication known as "opportunistic content based" in a discontinuous mobile ad hoc networks. This platform was developed using JAVA, known for its portability features, and runs in the Linux system.

The first objective of this internship is for studying and implementing the possibilities of porting DoDWAN in the Windows environment. And the second objective is for adding the feature of cryptography which can later be used to add security features to DoDWAN.

Keywords :

DoDWAN (Document Dissemination in Wireless Ad hoc Networks), opportunistic content based, discontinuous mobile ad hoc networks, portability, cryptography and security.

RÉSUMÉ

DoDWAN (Document Dissémination in Wireless Ad hoc Network) est une plate-forme intergicielle qui a été créé par l'équipe CASA du laboratoire VALORIA, qui permet de supporter le modèle de communication dit "opportuniste basé contenus" dans un réseau mobile ad hoc discontinu. Cette plate forme a été développée en JAVA pour sa caractéristique de portabilité, et fonctionne sur le système Linux.

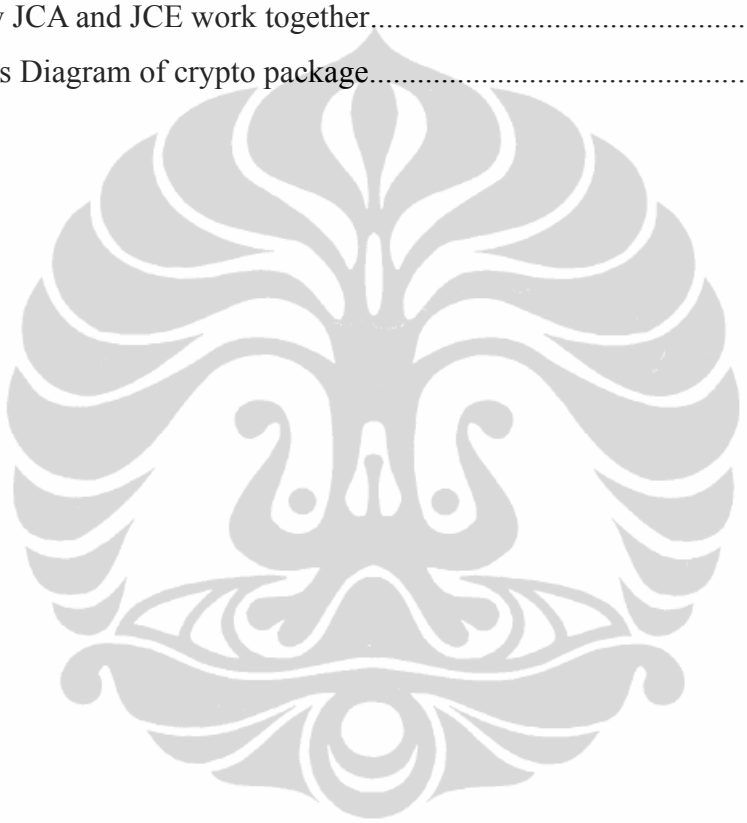
Le premier objectif de ce stage est d'étudier et de mettre en œuvre les possibilités de portage de la plate-forme intergicielle DoDWAN sur le système Windows. Le deuxième objectif est d'ajouter la fonctionnalité de chiffrement qui peut ensuite être utilisé pour ajouter des fonctionnalités de sécurité sur cette plate-forme.

Mot-clés:

DoDWAN (Document Dissémination in Wireless Ad hoc Network), opportuniste basé contenus, réseaux mobiles ad hoc discontinu, la portabilité, la cryptographie et la sécurité.

Table of Contents

Index of Figure



Index of Tables

1. Introduction

Ambient computing is one of the domain that develops rapidly in recent years and might be considered as a key in the future where people doing many amazing things by “using” computational devices and systems simultaneously, and may not necessarily even be aware that they are doing so. The goal of this technology is to achieve seamless integration between digital environments and physical world which provides the users more natural ways of interacting with their environment without being aware of using services of neighboring devices and systems.

The reason behind this rapid development is the emergence of new distributed architectures based on the computational devices (traditional computing units (such as servers and desktop workstations) or portable, mobile devices (such as laptops, personal digital assistants, or smart-phones), as well as embedded devices (such as sensors, actuators, control units, etc)), and the networks with short or long range wireless connection (such as UMTS, Blue-tooth, Wi-Fi, RFID, etc).

The CASA group is member of the VALORIA laboratory in the Université de Bretagne Sud that aims at proposing methods, models, and software tools to help design, deploy, and run services in ambient computing environments. One of their research is developing a middle ware platform for supporting the application and services in discontinuous mobile ad hoc networks. This platform is called DoDWAN (Document Dissemination in Wireless Ad hoc Networks) that supports the model of communication known as "opportunistic content based" in a discontinuous mobile ad hoc network. In this internship, I was given the opportunity by CASA group to be involved in the development process of the DoDWAN middle ware platform.

This internship consists of several sections. The first part of this report is devoted to the presentation of the VALORIA laboratory, CASA group and DoDWAN middle ware project. Then it continues with the description of the conducted

research activity in the internship. And finally, it ends with the conclusion and some perspective about future development.

1.1. Presentation of the VALORIA

VALORIA is the computer science laboratory at Université de Bretagne Sud. It develops research activities in the scope of Ambient Intelligence (AmI), addressing research and development topics along three complementary research activities:

- Interaction and Intelligence: This first activity aims at providing end-users with technological, innovative means for greater user-friendliness, more efficient services support and user-empowerment, while contributing to user-friendly, dependable, adaptive and non-intrusive hardware/software environments.
- Software Architecture: The second activity is dedicated to architecting/refining, testing/re-factoring and maintaining/evolving dynamic, distributed, mobile and context-aware systems considered as the background support to ambient computing.
- Middle ware: The third activity focuses on providing middle ware for distributed mobile and communicating systems as a support to ubiquitous and pervasive computing.

VALORIA is structured in four teams:

- ARCHWARE (software architectures);
- CASA (platforms for mobile computing);
- SEASIDE (SEarch, Analyse, Simulate and Interact with Data Ecosystems);
- RIMH (robotics and multimodal interactions towards disability).

The next part of this document will give the brief description about CASA group (**C**omposants **A**daptables sur **S**upports **A**daptables) and one of his project called DoDWAN.

1.2. Presentation of CASA Group

CASA is one of the research group in the VALORIA laboratory. The research activity of CASA group aims at proposing methods, models and software tools for the design, the deployment and the execution of applicative services in ambient computing environments. Such environments are generally characterized by the heterogeneity of the devices involved and by the dynamic nature of the available resources on these devices. The objective of CASA is to design software components that are aware of their environment so as to adapt to run-time conditions at deployment time and at execution time.

Ambient computing also implies the deployment –coordinated or not– of multiple digital devices equipped with interfaces for short range transmissions. In this perspective, CASA group mainly studies models of communication between mobile devices based on the ad hoc mode. Their approach consists in applying the DTN (Delay-Tolerant Networking) concept to the specific case of ad hoc networks. This approach allows supporting the strong constraints faced in this kind of network (low density, low connectivity, high volatility and mobility, etc.).

Lastly, CASA also investigate the use of the component and service-oriented approach for developing distributed ambient computing applications. Their work is concerned by the deployment of component-based and service-based applications on mobile devices interacting in an ad hoc network, and by the implementation of ubiquitous distributed services in this kind of environment.

1.3. Presentation of DoDWAN Project

Content-based communication is a style of communication where information flows towards interested hosts (receivers) rather than towards specifically set destinations. Receivers will specify the kind of information they are interested in, without regard to any specific source, while senders simply send information in

the network without addressing it to any specific destination. In fully connected wired networks, content-based communication can be achieved by constructing an underlying communication system, whose role is to forward each piece of information from its sender to all interested hosts. This system is usually organized as a logical, content-driven routing infrastructure, which itself can be implemented as an overlay network that covers the whole physical point-to-point network.

In the other side, the growth of laptops and others portable device such as PDA or smart phone that equipped with 802.11/Wi-Fi wireless networking have made MANETs a popular research. MANETs (Mobile Ad hoc NETworks) are self configuring infra structureless network of mobile device connected by wireless link. Each device in a MANET is free to move independently in any direction, and will therefore change its links to other devices frequently. Each must forward traffic unrelated to its own use, and therefore be a router. The primary challenge in building a MANET is equipping each device to continuously maintain the information required to properly route traffic. Such networks may operate by themselves or may be connected to the larger Internet. Many academic papers evaluate protocols and their abilities, assuming varying degrees of mobility within a bounded space, usually with all nodes within a few hops of each other. Different protocols are then evaluated based on measure such as the packet drop rate, the overhead introduced by the routing protocol, end-to-end packet delays, network throughput etc.

If we want to implement content-based communication in MANET the problem will appears, because a MANET can become disconnected when, for example, the mobile hosts that compose the network are very sparsely or irregularly distributed. The whole network then appears as a collection of distinct ``islands''. Communication between hosts that belong to the same island is possible, but no temporary communication is possible between hosts that reside on distinct islands.

In this disconnected MANET, the absence of end-to-end connectivity between distinct islands will make the content based communication using fully connected approach will hard to applied.

In such conditions, a method must be devised in order to bridge the gap between non connected parts of the network. Delay-tolerant networking is an approach that can help with that respect. In a delay-tolerant network, a message can be *stored* temporarily on a host, in order to be *forwarded* later by this host when circumstances permit. If the network includes mobile hosts---or if all hosts are mobile---, then mobility becomes an advantage, as it makes it possible for messages to propagate network-wide, using mobile hosts as *carriers* that can move between remote---and possibly non-connected---fragments of the network. In a disconnected MANET such as that shown in Figure 1, people moving between buildings (or between different parts of a building) can thus contribute to disseminate information between non-connected fragments of the MANET. Figure 2 shows a typical example, where the laptop of a user moving between two groups of users can contribute to carry information between these two groups.

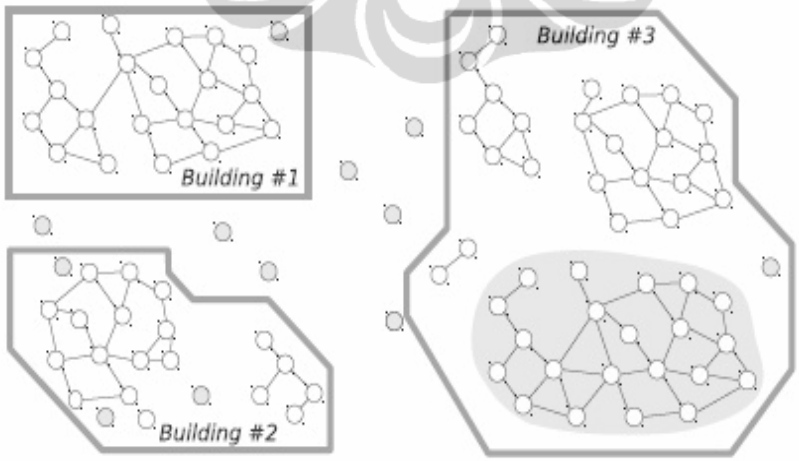


Figure 1. Example of Disconnected MANETs

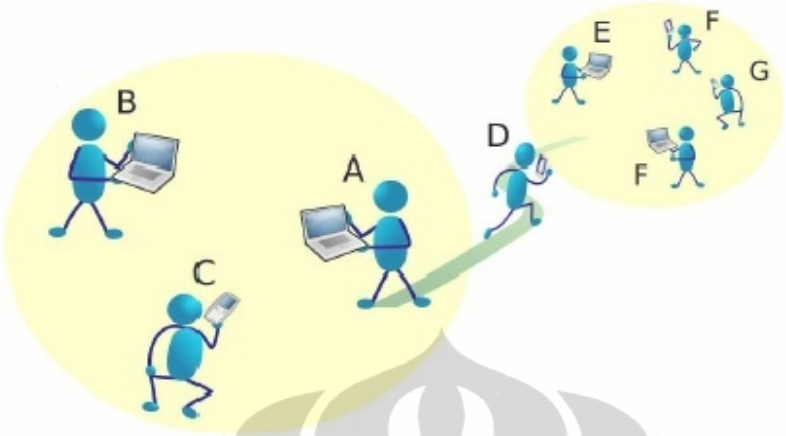


Figure 2. Detail of figure 1, showing how users moving in the campus can carry information between connectivity islands.

DoDWAN (stands for Document Dissemination in disconnected Wireless Ad hoc Networks) is the middle ware platform designed by CASA group in order to support content-based information dissemination in disconnected mobile ad hoc networks. DoDWAN is distributed under the terms of the GNU General Public License. It provides high-level application services with means to publish and subscribe for structured pieces of information we refer to as *documents*, using a dedicated API.

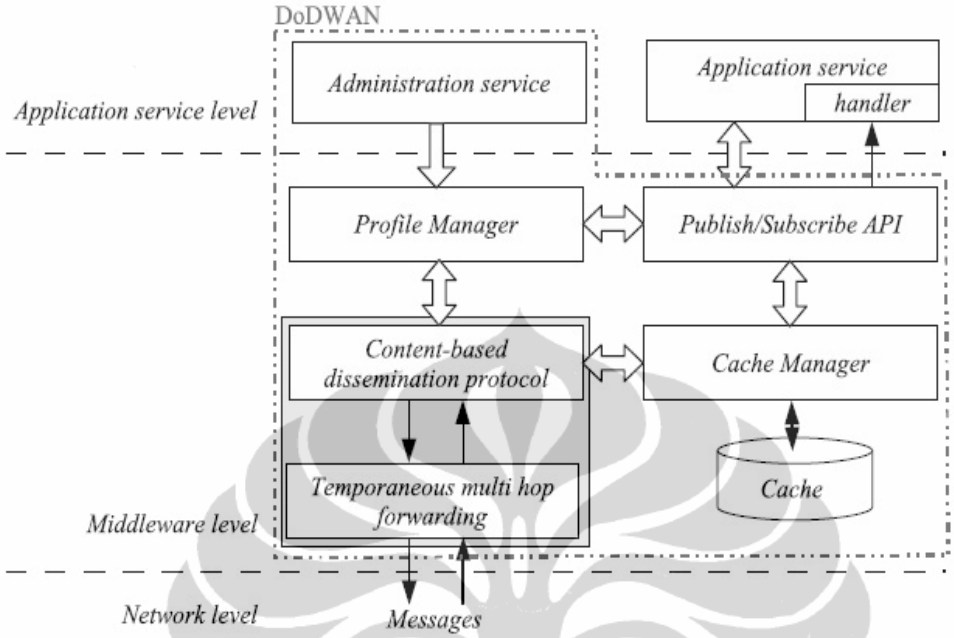


Figure 3. Architecture of the communication platform

This platform is not just simulation code: it has been fully implemented in Java, and is now being used for developing effective application-level software. Details information about DoDWAN can be read in the article [1] publish by CASA group.

1.4. Objectives of the Internship

As mentioned in the part of introduction above, CASA group gave me the opportunity to join their research group. The general objective of this internship is to learn and involve in the development process of the DoDWAN middle ware platform. CASA group wants to develop this middle ware by adding some functionalities. DoDWAN was developed using JAVA and running in the Linux operating system. CASA group has the interest about the possibilities of porting this middle ware in Windows system. And also because DoDWAN is running under "open" wireless ad hoc network, it lies the problem of security. The CASA group wants to add the cryptography functionality to overcome this security problem.

The objectives of this internship are :

1. Porting DoDWAN in Windows system
2. Adding cryptography service in DoDWAN middle ware platform.

1.5. Project Planning

The following are project plannings to achieve the objectives for the internship :

1. Task Identification

   a) Bibliographic study about DoDWAN : this task is for studying the DoDWAN middle ware platform and knowing all the technology behind this middle ware and how it works by reading documents published by the CASA group.
   b) DoDWAN testing in Linux environment : this task is for testing all the functionalities of the DoDWAN platform that had been developed and testing it in Linux operating system for achieving a better understanding about how DoDWAN works.
   c) DoDWAN porting in Windows environment : this task is for porting DoDWAN in Windows environment, and testing all DoDWAN functionalities in Windows, not only the middle ware platform but also all the DoDWAN application services that had been developed by the CASA group, such as the service of presence, file sharing, mail, chat, news and game that utilize the DoDWAN platform.
   d) Bibliographic study about cryptography in JAVA : this task is for studying the cryptography functions and implementation in JAVA that can be added as a service in DoDWAN.
   e) Analysis for cryptography service in DoDWAN : this task is for creating the specification requirements for the cryptography service in DoDWAN.
   f) Development for cryptography service in DoDWAN : this task is the development process for the cryptography service in DoDWAN.

g) Internship report : this task is for creating internship report.

2. Workload Estimation

Table 1. Workload Estimation

Task	Estimation	In weeks
Bibliographic study about DoDWAN	5 days	1 week
DoDWAN testing in Linux	10 days	2 weeks
DoDWAN porting in Windows	25 days	5 weeks
Bibliographic study about cryptography in JAVA	10 days	2 weeks
Analysis	20 days	4 weeks
Development	30 days	6 weeks
Internship report	10 days	2 weeks
Total	110 days	22 weeks

3. Workload Planning

The task planning for this internship in the period of 26 January 2011 until 1 July 2011 :

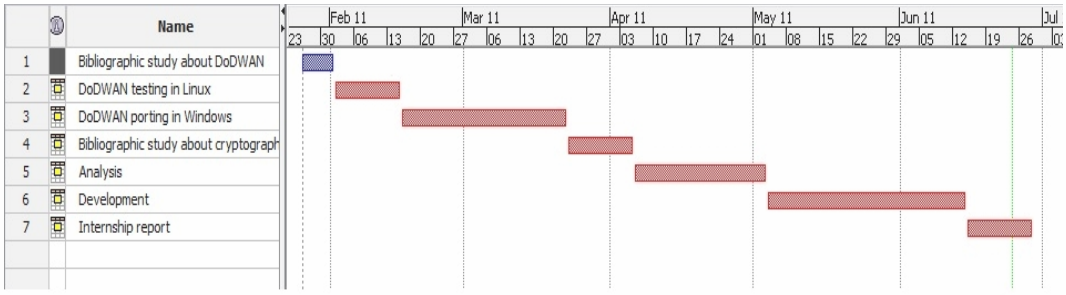


Figure 4. Task Planning for the internship

For monitoring the internship, there will be meetings with the supervisor

and other member of CASA group. The meetings are for giving the information about the progress of the internship, resolving any problems, setting the main objectives of each task, and prioritizing the work to finish.

2. Analysis

As mentioned in the project objectives above, this internship has two objectives, the first is to porting DoDWAN in Windows, and the second is to add the cryptography functionality in DoDWAN. This section contains the analysis that has been done to achieve those two goals.

2.1. Porting DoDWAN in Windows

One of the objectives to be achieved in this internship is to porting the DoDWAN middle ware in Windows systems. DoDWAN originally was created using the Java programming language for Linux operating systems. We have a huge advantage when it comes to the process of porting DoDWAN to Windows operating system, because Java is a programming language known for its portability feature, we only need to compile once, and it will run everywhere, as long as that computers have the Java Runtime Environment. Here are some points that may need some attention for this process :

1. The software version compatibility used in both operating system :

   We must ensure that Java run time environment in Linux and Windows system is the same and using the minimum version needed to run the DoDWAN middle ware platform. For running the other service that use DoDWAN platform called DoDWAN applications, such as presence, file sharing, mail, chat, news and game, we need the OSGI platform. We must ensure that the version of this OSGI platform running in Windows is the same as the one used in Linux system. And also, if the DoDWAN applications running in Linux System are using other applications or software, we must search the alternative application that also runs in Windows system.

2. File access in both operating system :

   There are differences about how to access files on Windows and linux operating systems. In Linux system we use "/" separator to access the file or file path, while in Windows system we use "\". We must look at the source code of DoDWAN, and notice how the file access is implemented.

We must ensure that no files are accessed in an absolute path (using "/"). If that method of file access exists, we have to change that by using a Java plateform independent features for file access.

3. Mastering Wi-Fi ad hoc configuration :

   DoDWAN runs in disconnected wireless ad hoc networks, and it needs some mechanism and configuration in order that the mobile machines can connect automatically in this DoDWAN network. In the Linux system this function is implemented by using specific channel number of the ad hoc network. However, to make this "connect automatically" function in Windows system will be complicated, since by default Windows doesn't allow to connected automatically to a wireless ad hoc network because of security reasons. We must mastering Wi-fi ad hoc configuration in Windows to find some mechanism to make Windows connect automatically in the ad hoc network.

Based on above analysis, here are some tasks to do, to achieve the first objective :

1. Test DODWAN with Wi-Fi under Linux
2. Review the init scripts and the configuration files
3. For porting DoDWAN in Windows, we must do :
   a) Preliminary tests DoDWAN in Windows
   b) Modification the DODWAN code for accessing the file system
   c) Mastering Wi-Fi ad hoc configuration

2.2. Cryptography Service for DoDWAN

The other objective for this internship is to provide cryptography functionality in DoDWAN middle ware which can later be used to add security features of the DoDWAN. The DoDWAN middle ware is running under Wi-Fi ad hoc network with "open" condition. It means that everyone that have the wireless device that support ad hoc connection, can connect in the DoDWAN network. If we want to add the security features in this wireless ad hoc network there are many options

for making this network secure.

Cryptography is one of the solution that can make this DoDWAN middle ware platform secure. According book [8], we must realize that a cryptographically enabled program is not necessarily a secure one. Without a carefully planned and constantly secure strategy, cryptography won't help much. Correctly used, cryptography provides these standard security features:

- Confidentiality assures that data cannot be viewed by unauthorized people.
- Integrity assures that data has not been changed without our knowledge.
- Authentication assures that people we deal with, are not imposters.

DoDWAN was developed using JAVA programming language. Fortunately, the JAVA language continues to offer a computing platform that swells with cryptographic function. This functionality is split beween two different libraries, the JAVA Cryptography Architecture (JCA) and the JAVA Cryptography Extension (JCE). Figure 5, taken from book [5], shows how the various parts work together. Application code is written that calls the appropriate JCE/JCA API classes; these in turn invoke the classes in a provider that provides implementations for the JCE/JCA service provider interface (SPI) classes. The classes then invoke the internal code in the provider to provide the actual functionality requested.

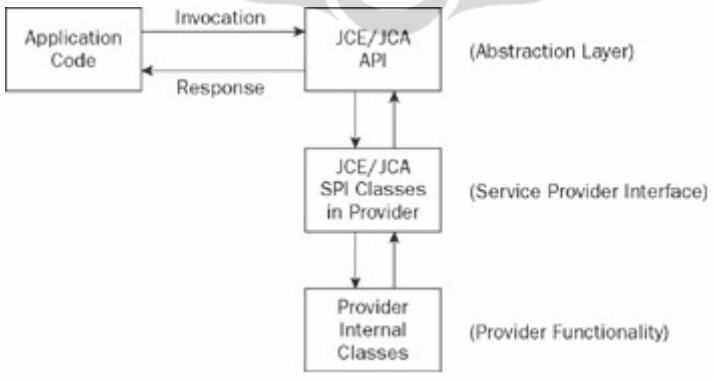


Figure 5. How JCA and JCE work together

The Java Security API is a set of packages that are used for writing secure programs in Java. In particular, the classes and interfaces in the following packages are part of the Security API:

- java.security
- java.security.cert
- java.security.interfaces
- java.security.spec
- javax.crypto
- javax.crypto.interfaces
- javax.crypto.spec

Here are some points that may be considered for implementing cryptography in DoDWAN platform :

1. Which Java Security API to be used

   The java.security and javax.crypto packages have classes and interfaces that represent the cryptographic concepts. The following table 2 summarizes the cryptographic concepts represented in the classes included in JCA and JCE.

   Table 2. Cryptographic Classes Concept in JAVA

No	Class or Interface	Description
1	java.security.cert.Certificate	A cryptographic certificate
2	javax.crypto.Cipher	A cipher
3	java.security.Key , java.security.PrivateKey , java.security.PublicKey , javax.crypto.SecretKey	A key, used for signing or encryption
4	javax.crypto.KeyAgreement	A secret key exchange protocol
5	java.security.KeyFactory	Translates public and private keys from one format to another
6	javax.crypto.KeyGenerator	Creates keys for symmetric ciphers

No	Class or Interface	Description
7	java.security.KeyPairGenerator	Creates pairs of public and private keys for signing or encryption
8	javax.crypto.Mac	A Message Authentication Code (MAC)
9	java.security.MessageDigest	A cryptographic hash function
10	javax.crypto.SecretKeyFactory	Translates secret keys from one format to another
11	java.security.SecureRandom	A cryptographically strong random number engine
12	java.security.Signature	A digital signature

In the DoDWAN platform we need the functionality for encrypting and decrypting message, symmetric and asymmetric, in order that the transmitted document in the wireless ad hoc network become more secure. Everyone in the DoDWAN ad hoc network can receive the encrypted documents, but only the user with the right key can decrypt and read that documents. Based on sumarization in table 2 and the functionality of encrypting message, we will ned the class or interface number 1, 2, 3, 5, 6, 7 and 10.

2. Key Management

According book [6], Cryptography can be a two-edged sword, as much as it helps, it has the potential to hurt. For example, if we encrypt the secret message, and then we lose the key that we used to encrypt it, that message may be irretrievable. And also, how can we store all the key (especially secret key) safely? Or, Is there a way of persisting a secret key for longer-term use? These are some question that appeared when we want to manage the key. That is why we need some mechanisme for tracking and managing secret keys, key pairs, and digital certificates in DoDWAN middle ware platform.

According book [5], JAVA has the keystore facilities in the `KeyStore` class. Primarily, Java `KeyStore` objects are used to store private keys and their associated certificates. The `java.security.KeyStore` class was introduced in JDK 1.2. The KeyStore API supports the persisting of three types of entries:

- **Private keys.** Private keys can be saved with their associated certificate chains. In this entry we can save the Key Pair (Private and Public key) in the Keystore. In most cases they can also be individually password protected.
- **Symmetric keys.** Not all type Keystore support this secret key use in symmetric cryptography. Only keystore with the type SUN JCEKS can save this secret key. This secret key is save to the keystore when the entry is not certificate. Where the saving of symmetric keys does work they can be individually password protected.
- **Certificates.** These are the certificates used to create public key. Ordinarily we will have obtained them from a third party and verified their authenticity through channels other than those we use for validating certificates that exist within certificate paths. But, in the DoDWAN wireless ad hoc, there isn't any server that can play a role as a Certificate Authority. So in these implementation we use the self sign certificate to generate certificate from public key.

There are three basic keystore types that ship with the JDK, which also have some minor variations:

- **JCEKS**—This is a Sun format type that was introduced with the JCE. In addition to being able to contain private keys and certificates, it can also contain symmetric keys. It differs from the `JKS` in that the encryption used to protect private keys is based on Triple-DES, which is much stronger than the algorithm used in the `JKS`. Aliases are case-

insensitive.

- **JKS**—This is the original Sun format keystore type. It will only contain private keys and certificates, and aliases are case-insensitive. There is also a variation on it, `CaseExactJKS`, which recognizes the aliases with the same spelling but different case.
- **PKCS12**—Aversion of the format defined in RSA Security's PKCS #12. Up till JDK 1.5 this type was read-only, but we can now write them as well. Aliases are case-insensitive. The store cannot be used to store trusted certificates.

There are other type of keystore beside this basic keystore, for example the keystore implementation by IBM and keystore implementation in Bouncy Castle library. But in this internship, we just need to save the secret key and key pair, so simply by using the standard type of JCEKS, we can do the key management.

3. Implementation

This section contains the implementation that has been done based on the analysis section above to achieves the objectives of the internship. Generally, there are two section of implementation that have been conduct. The first is about the implementation of porting DoDWAN in Windows and the second is about the implementation of adding cryptography functionality in the DoDWAN platform.

This section is consists of 3 part :

1. Development environment
2. The result of porting DoDWAN in Windows
3. The implementation of DoDWAN crypto package

3.1. Development Environment

In this internship, there are some tools that been used for testing and developing the DoDWAN platform :

1. IDE Eclipse

   According to their website (http://www.eclipse.org/org/), Eclipse IDE is an Integrated development that come from Eclipse open source community, whose projects are focused on building an open development platform comprised of extensible frameworks, tools and run times for building, deploying and managing software across the life cycle. We use eclipse IDE as development environment to develop the DoDWAN middle ware platform since Eclipse IDE has many advantages, including easy integration with Maven. In this internship we use Eclipse Helios version 3.6.1.

2. Maven

   According to their website (http://maven.apache.org/), Apache Maven is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information.

   Maven is a tool that can now be used for building and managing any Java

-based project. Maven's primary goal is to allow a developer to comprehend the complete state of a development effort in the shortest period of time. In order to attain this goal there are several areas of concern that Maven attempts to deal with:

- Making the build process easy
- Providing a uniform build system
- Providing quality project information
- Providing guidelines for best practices development
- Allowing transparent migration to new features

In this internship we used apache-maven-2.2.1 version.

3. Felix

   According to their website (http://felix.apache.org/site/index.html), Apache Felix is a community effort to implement the OSGi R4 Service Platform and other interesting OSGi-related technologies under the Apache license. The OSGi specifications originally targeted embedded devices and home services gateways, but they are ideally suited for any project interested in the principles of modularity, component-orientation, and/or service-orientation. OSGi technology combines aspects of these aforementioned principles to define a dynamic service deployment framework that is amenable to remote management.

   In this internship, for testing the DoDWAN and DoDWAN application, we use Felix Framework Distribution 1.8.0 and 3.0.8 version. We use the 1.8.0 version because that was the version used in the Linux distribution. And we also used the 3.0.8 version because it was the latest version when this internship is conducted.

4. SVN

   According to this site (http://svnbook.red-bean.com/en/1.5/svn.intro.whatis.html), Subversion is a free/open source

version control system. That is, Subversion manages files and directories, and the changes made to them, over time. This allows to recover older versions of our data or examine the history of how our data changed. In this regard, many people think of a version control system as a sort of "time machine."

Subversion can operate across networks, which allows it to be used by people on different computers. At some level, the ability for various people to modify and manage the same set of data from their respective locations fosters collaboration. Progress can occur more quickly without a single conduit through which all modifications must occur. And because the work is versioned, we need not fear that quality is the trade-off for losing that conduit—if some incorrect change is made to the data, just undo that change.

In this internship, for Windows system we can use the Turtoise SVN with graphical user interface or Slik subversion for text based SVN.

3.2. Porting DoDWAN in Windows Results

Here are some task for testing and developing DoDWAN that have been done for achieving the premier objective of this internship :

3.2.1. DoDWAN Test Under LINUX

For testing DODWAN middle ware on Linux system, we use computers with Linux Ubuntu 10.10 operating system. DoDWAN uses Java programming language, so we install Sun JDK and Sun JRE on this system. We do this test in two ways, the first is the manual way (we just install the DoDWAN platform) and the second is the automatic way (we install DoDWAN platform and all the application that uses this platform, presence, file sharing, mail, news, chat and game).

- For the first test: We download DoDWAN, then follow the tutorial of the network configuration steps, and finally, test all the DODWAN

functionality (using daemon and using java source code sample). All the DoDWAN packet, network configuration and functionality test tutorial were available at the DoDWAN website, http://www-valoria.univ-ubs.fr/CASA/DoDWAN/index-en.html .

Conclusion: DODWAN run well on a wired/Wi-Fi network using ipv4 and ipv6 in Linux Ubuntu 10.10 system.

- For the second test, We download the DODWAN middle ware and its applications in the repository provided by the team CASA of VALORIA laboratory. First, in the file /etc/apt/sources.list, we add the following repository :

```
http://www-valoria.univ-ubs.fr/CASA/DODWAN-EXPE/debian
dodwan main
```

and then, install DODWAN using the command :

```
apt-get install dodwan
```

Conclusion: DODWAN package with its applications service run well on a wired/Wi-Fi network using ipv4 and ipv6 in Linux Ubuntu 10.10 system.

3.2.2. DoDWAN Test Under Windows

In this section, We perform several tests to DODWAN middle ware on computers that use Microsoft Windows 7 operating system.

A. Preliminary Tests

We have done 3 type of preliminary test in the Windows system. The first is for testing the DoDWAN platform, the second is for testing all the bundle service that

use DoDWAN platform and the final test is for testing the source code of DoDWAN platform and DoDWAN application service by compiling and testing in the Windows system.

For the first test, We download the dodwan-1.0.4.jar bundle and then copy this bundle in the home directory. We can copy this file to any directory in Windows system. But in this preliminary test, we try to make DoDWAN folder in Windows home directory the same as the DoDWAN distribution for Linux which is in the home directory of the user.

To run the DoDWAN (using daemon and Java API), we create batch script to execute DoDWAN daemon or DoDWAN JAVA API. Here are the batch script for running the DoDWAN using daemon (create this .bat file in this following path : `C:\Users\user\DoDWAN\home\bin`) :

```
REM scriptdaemon
@ECHO OFF
java -cp %userprofile%\DoDWAN\home;%userprofile%\DoDWAN\home\lib\
dodwan-1.0.4.jar casa.dodwan.run.dodwand
```

And here's the batch script for running DoDWAN using Java API (create this .bat file in the following path : `C:\Users\user\DoDWAN\home\samples`). There are two script, one for subscriber and the other for publisher :

```
REM scriptsubscriber
@ECHO OFF
javac -cp %userprofile%\DoDWAN\home\lib\dodwan-1.0.4.jar
sampleSubscriber.java
java -cp %userprofile%\DoDWAN\home;%userprofile%\DoDWAN\home\lib\
dodwan-1.0.4.jar samples.sampleSubscriber
```

```
REM script publisher
@ECHO OFF
javac -cp %userprofile%\DoDWAN\home\lib\dodwan-1.0.4.jar
samplePublisher.java
java -cp %userprofile%\DoDWAN\home;%userprofile%\DoDWAN\home\lib\
dodwan-1.0.4.jar samples.samplePublisher
```

Here is the example where we place the DoDWAN file in the Windows system :

Table 3. DoDWAN file and location path in Windows

File	Path File
`dodwan-1.0.4.jar`	`C:\Users\user\DoDWAN\home\lib`
`dodwan.defaults`	`C:\Users\user\DoDWAN\home\conf`
`samplesubscriber.java` `samplepublisher.java`	`C:\Users\user\DoDWAN\home\samples`
`start_dodwan.bat`	`C:\Users\user\DoDWAN\home\bin`
`samplesubscriber.bat` `samplepublisher.bat`	`C:\Users\user\DoDWAN\home\samples`

Conclusion : all the functionality of the dodwan-1.0.4.jar runs well in Windows 7 system.

For the second test, We tried to run several bundles that available in the repository https: / / www-valoria.univ-ubs.fr/svn/casa/devlpts/dodwan-home/bundle/, using OSGI framework Felix 1.8.0 and 3.0.8 version. The version 1.8.0 is the framework that used in the DoDWAN Linux distribution, while the the version 3.0.8 is the latest version of the Felix when this report is made. We tested this following bundle : chatwan-1.0.1.jar, dodwan-1.0.4.jar, fishwan-1.0.4.jar, fishwangui-1.0.4.jar, gamewan-1.03.jar, gamewangui-1.0.2.jar, mailwan-1.0.2.jar, newswan-1.0.2.jar and presencegui-1.0.4.jar.

Conclusion : All the bundle run well in Windows 7 system.

And for the final test, We downloaded all the source code of each bundle in the repository, and then We re-compiled all the source code using Apache-Maven 2.2.1 version and also using the Eclipse IDE 3.6.1 version. The first source code

that we download in the DoDWAN SVN repository is the 1952 revision. But apparently, there was the improvement (the 2117 revision) in the DoDWAN SVN directory at Friday, 11 March 2011. We test by re-compiling this two version of the DoDWAN (revision 1952 and revision 2117).

Here are the results of the 1952 revision :

- DoDWAN (rev 1952) → OK
- presencegui (rev 1952) → Warning
- fishwan (rev 1952) → Error
- fiswangui (rev 1952) → Warning
- chatwan (rev 1952) → OK
- mailwan (rev 1952) → OK
- newswan (rev 1952) → Error
- gamewan (rev 1952) → OK
- gamewangui (rev 1952) → OK

Table 4. The results of re-compiling the DoDWAN source code (revision 1952) in the Windows system.

Bundle	Status	Information	Note

newswan (rev 1952)	Error	`[INFO] Compilation failure`
		
		`C:\stage\dev\DoDWANapp\newswanrev2026\src\main\java\org\cooldevtools\coffee_server\services\nntp\protocol\NntpService_POST.java:[63,23] cannot find symbol`
		`symbol : class MimeUtility`
		`location: package casa.DoDWAN.util`
		
		`C:\stage\dev\DoDWANapp\newswanrev2026\src\main\java\org\cooldevtools\coffee_server\services\nntp\protocol\NntpService_PO`
			1.SVN from Windows7 failed because there are two files that considered to have the same name, which is: newswan.java and Newswan.java
			
			Analysis: Windows 7 is considered this two files is the same file
			
			Solution: Rename a File newswan.java be newswans.java

Bundle	Status	Information	Note
		ST.java:[1271,12] cannot find symbol symbol : variable MimeUtility location: class org.cooldevtools.coffee_server.services.nntp.protocol.NntpService_POST C:\stage\dev\DoDWANapp\newswanrev2026\src\main\java\org\cooldevtools\coffee_server\services\nntp\protocol\NntpService_POST.java:[1283,12] cannot find symbol symbol : variable MimeUtility location: class org.cooldevtools.coffee_server.services.nntp.protocol.NntpService_POST	2. Compile error Analysis: casa.dodwan.util.MimeUtility.java file does not exist in the src folder (although the distribution of the dodwan bundle-1.0.3.jar has already had a class MimeUtility). Solution: add the file MimeUtility.java in the folder src
fishwan (rev 1952)	Error	[INFO] Compilation failure C:\stage\dev\DoDWANapp\fishwan\src\main\java\casa\DoDWANapp\fishwan\FishwanConsole.java:[36,22] cannot find symbol symbol : class XmlAttributeParser location: package casa.DoDWAN.xml C:\stage\dev\DoDWANapp\fishwan\src\main\java\casa\DoDWANapp\fishwan\FishwanConsole.java:[202,13] cannot find symbol symbol : variable XmlAttributeParser location: class casa.DoDWANapp.fishwan.FishwanConsole	1. Compile error Analysis: XmlAttributeParser.java file does not exist in the src folder (although the distribution of the dodwan bundle-1.0.3.jar has already had a class XmlAttributeParser Solution: add the file XmlAttributeParser.java in the folder src
fiswangui (rev 1952)	Warning	[INFO] Compiling 75 source files to C:\stage\dev\DoDWANapp\fishwangui\target\classes [WARNING] C:\stage\dev\DoDWANapp\fishwangui\src\main\java\	Not Error just Warning sun.swing.plaf.synth.DefaultSynthStyle is Sun proprietary API and may

Bundle	Status	Information	Note
		casa\DoDWANapp\fishwangui\gui\FishwanGuiLookAndFeel.java:[36,27] sun.swing.plaf.synth.DefaultSynthStyle is Sun proprietary API and may be removed in a future release [WARNING] C:\stage\dev\DoDWANapp\fishwangui\src\main\java\casa\DoDWANapp\fishwangui\gui\FishwanGuiLookAndFeel.java:[74,33] sun.swing.plaf.synth.DefaultSynthStyle is Sun proprietary API and may be removed in a future release [WARNING] C:\stage\dev\DoDWANapp\fishwangui\src\main\java\casa\DoDWANapp\fishwangui\gui\FishwanGuiLookAndFeel.java:[75,22] sun.swing.plaf.synth.DefaultSynthStyle is Sun proprietary API and may be removed in a future release	be removed in a future release
presencegui (rev 1952)	Warning	[WARNING] Embed-Dependency: clause "jcalendar" did not match any dependencies	Remove the dependency of jcalendar class.

And here are the results of the 2117 revision (after 11 March 2011) :

- dodwan (rev 2117) → OK
- presencegui (rev 2117) → Warning
- fishwan (rev 2117) → OK
- fiswangui (rev 2117) → Error
- chatwan (rev 2117) → OK
- mailwan (rev 2117) → OK

- newswan (rev 2117) → Error
- gamewan (rev 2117) → OK
- gamewangui (rev 2117) → OK

Table 5. The results of re-compiling the DoDWAN source code (revision 2117) in the Windows system.

Bundle	Status	Information	Note
Presencegui (rev 2117)	Warning	`[WARNING] Embed-Dependency: clause "jcalendar" did not match any dependencies`	Remove `jcalendar` dependency in `pom.xml` file (in line 48), because we don't need this package for presencegui bundle.
Fiswangui (rev 2117)	Error (running)	`g! start 11 FishwanGuiActivator.start() : Starting Fishwangui 1.0.4 g! fishwangui#FishwanGui.setLookAndFeel(): loading casa.dodwanapp.fishwangui.gui.FishwanGuiLookAndFeel fishwangui#FishwanGuiLookAndFeel.getDefaults(): default font size = 10 fishwangui#MainFrame() : Failed getResourceAsStream(/casa/dodwanapp/fishwangui/language/.languages) fishwangui#MainFrame() : failed supportedLanguages.load(null)`	The folder language which consist of the source for language class didn't exist in the source code repository.
Newswan (rev 2117)	Error (Windows file system)	SVN commit in Windows 7 failed because there are two files, `newswan.java` and `Newswan.java`, that considered having the same name.	Analysis: Windows 7 is regarded both file as the same file. Solution: rename a file → `Newswan.java` be `NewswanS.java`, and after changed the name of this class, then made the adjustments to the source code files.

Conclusion : Although all bundle run well as we test in the second test, but when we re-compiled the source code, there is some error in the sorce code that needs to

be fix.

Here are the changes that made in the dodwan-1.0.3.jar package (1952 revision) :

1. File: casa.dodwan.util.RemoteConsole.java (line: 25)

   Reason : wrong package

   Original:

```
casa.dodwan.transmission package;
```

   Changes :

```
casa.dodwan.util package;
```

2. File: casa.dodwan.pubsub.ReceptionHistory.java (line:57)

   Reason : in the Windows system we can not create bak file, if the bak file is exist. To fix that we check the existence of the bak file, if it exist we delete this existing bak file, if it isn't we create the new bak file.

   Original:

```
boolean renamed = file.renameTo (bakFile);
if (renamed) {
 System.out.println ("ReceptionHistory(); failed to rename"
 File.getAbsoluteFile + () + "to"
 BakFile.getAbsoluteFile + () + ". Bak");
 return;
}
```

   changement:

```
//Modified AT
if (bakFile.exists()) {
 bakFile.delete();
}

boolean renamed = file.renameTo (bakFile);
if (renamed) {
 System.out.println    ("ReceptionHistory (); failed to
rename"
 File.getAbsoluteFile + () + "to"
 BakFile.getAbsoluteFile + ());
 return;
}
```

```
/ / End modified AT
```

3. File: casa.dodwan.util.PersistentMap.java (line: 62)
   Reason : File access using absolute path (using "/").

   Original:

```
try {
      File file = new File (path_ + "/" + key);
      file.delete ();
} Catch (Exception e) {}
```

   changement:

```
// Modified AT
File file = null;
try {
 file = new File (path_, key.toString ());
 file.delete ();
} Catch (Exception e) {}
// End modified AT
```

Finally, we need some application for running DoDWAN application service in Linux System. For the presence and file sharing (presence GUI and Fishwan GUI) we used the application that build by CASA group using JAVA. It means there won't be any problem to run this application in Windows system. But other service need third party application. Mail and news need Thunderbird, chat needs Xchat, and game needs GNUChess. Here are the list application that needed for running DoDWAN application service in Linux system and their compatibility in Windows :

Table 6. Application compatibility

Linux	Windows
Presence GUI, Fishwan GUI	Compatible

Thunderbird	Compatible (thunderbird for Windows exist)
Xchat	Compatible (Xchat for Windows exist)
GNUChess	Not Compatible, Alternative : WinBoard

B. Modification the DODWAN code for accessing the file system

In the previous dodwan-1.0.3.jar version (revision 1952), there is some change in class (casa.dodwan.util.PersistentMap.java → line: 62), because this class accessing file system using "/", so we change this. But this class is removed in dodwan-1.0.4.jar version (revision 2117).

In general, no significant problems lies in access file method in both operating system, because although access files between Linux and Windows is different, Linux use "/" while Windows uses "\", access to the file system using "/" is also recognized in Windows (in Windows, access to file systems `C:\Users\user` can also be accessed with `C:/Users/user`). Moreover, the file access method in DoDWAN middle ware is well written, because it rarely uses an absolute path (we find only one class that use "/"). Even DoDWAN paltform using DoDWAN environment class for accessing file.

C. Mastering Wi-Fi ad hoc configuration

Setting a route for multicast networking

The command for subscribe persistently to the multicast address using ipv6 in Windows is:

```
PS C:\Windows\system32> netsh interface ipv6 add route
ff02::236:1:2:3/128 "Wireless Network Connection 2"

Ok.
```

The option `ipv6` is for using ipv6, `ff02::236:1:2:3/128` is the multicast address and `"Wireless Network Connection 2"` is the network interface address. If we want to use ipv4 the example command will be :

```
PS C:\Windows\system32> netsh interface ipv4 add route
236.1.2.3/32 "Wireless Network Connection 2"
```

```
Ok.
```

Connect Automatically

In the case of DoDWAN, sometimes we need to connect automatically in some wireless ad hoc network, for example : we want to connect to ad hoc network named “dodwan”. If we want Windows 7 connected automatically to an ad hoc network, it needs a little trick , since Windows 7 by default doesn't allow this because of the security reasons. First, create the profile file : profile.xml, and then create a batch script file in the Startup folder of Windows 7, in order that this batch file can be executed automatically each time Windows 7 start. The batch script will add profile.xml in the system and at the same time, we connected the device with a wireless ad hoc network using netsh. Here is the file example of the profile.xml:

```
<?xml version="1.0"?>
<WLANProfile
xmlns="http://www.microsoft.com/networking/WLAN/profile/v1">
      <name>DODWAN-Windows7</name>
      <SSIDConfig>
            <SSID>
                  <hex>444F4457414E2D57494E444F575337</hex>
                  <name>DODWAN-Windows7</name>
            </SSID>
            <nonBroadcast>false</nonBroadcast>
      </SSIDConfig>
      <connectionType>IBSS</connectionType>
      <connectionMode>manual</connectionMode>
      <autoSwitch>false</autoSwitch>
      <MSM>
            <security>
                  <authEncryption>
                        <authentication>open</authentication>
                        <encryption>none</encryption>
                        <useOneX>false</useOneX>
                  </authEncryption>
            </security>
      </MSM>
</WLANProfile>
```

And here is the batch script, and put this script in the startup folder :

```
@echo off
netsh wlan add profile filename=C:\stage\profile.xml
      interface="Wireless Network Connection 2"
```

```
netsh wlan connect ssid=DODWAN-Windows7 name=DODWAN-Windows7
interface="Wireless Network Connection 2"
```

In the example above, we create the ad hoc network with the SSID name "DODWAN-Windows7" in the profile.xml file, and then connect to this SSID using the batch script.

Wireless Ad Hoc Channel Configuration in Windows

Although the netsh command has many capabilities, but unfortunately, until this tutorial is created, we can not configure the specific channel number for our ad hoc wireless networks in Windows system. For operates, DoDWAN needs specific channel number of the ad hoc network, for example channel number "3". This functional depends entirely on the drivers supplied by vendors of our wireless network adapters. If the driver provides features to configure the channel number then we can apply it on DoDWAN. But if the driver do not provide this feature, here are some tricks that can be done.

The solution we used for this problem is to change the driver of the wireless device to use the specific channel. For example, wireless network interface that we used in this research is D-Link DWL-G122 and it has the channel number 1 by default. Windows 7 will automatically know this device and uses the generic driver netr7364.inf for this device. To force this device to use channel 3, we changed the driver netr7364.inf and changes the channel number 1 become 3.

```
...
;
; Parameters
;
RadioOnOff                  = "Radio On/Off"
Disable                     = "Disable"
Enable                      = "Enable"
IEEE802_11h                 = "IEEE802.11h"
SmartScan                   = "SmartScan"
CarrierDetect               = "Carrier Detect"

CHANNEL                     = "3"
WPS_DEVNAME_DEF_STR         = "Ralink Client"
```

If we use another wireless device, the steps to configure the number of channel is

substantially the same. First, we look for the driver for this device, and then changed the driver *. inf file and change the channel number by default.

3.3. Cryptography Service for DoDWAN Results

Based on requirement analysis above, in general this cryptography service must provide two main function, encrypt/decrypt message using symmetric/asymmetric cryptography and keys managements. Here are some functionality that must be provided by crypto package :

1. Cryptography functionality in DoDWAN console
   a) We must provide the method for encryption and decryption process using symmetric and asymmetric cryptography. And also the method for list all supported encryption and decryption algorithm and list all alias (owner) of the key. All this cryptographic service will be add in the CryptoService class.
   b) To make this CryptoService class accessible in the DoDWAN console, we must create the CryptoServiceConsole class.
2. Key Managements
   a) The method for generates keys, either secret key for symmetric cryptography or key pair for asymmetric cryptography must be provided in our crypto service package. This method will create keys based on the name of the algorithm (either symmetric or asymmetric) and the size of the key. This method will be add in MyKeyGenerator class.
   b) We must provide the method for make persistent keys (secret key and key pair) by saving in the key file. This key file will have two attributes, the first is the alias (owner) of the key and the second is the value of the key. This method also has the functionality to save and load keys from the key store. All this functionality will be add in MyKeyExtractor class.
   c) The method for key managements using JAVA keystore class. This

keystore management class will provide all the service regarding the key management in keystore class, for example set and get keys (secret key or key pair), list all alias (owner) of the key and remove alias from the key store.

Based on all the functionality list above, we make Class Diagram of crypto package:

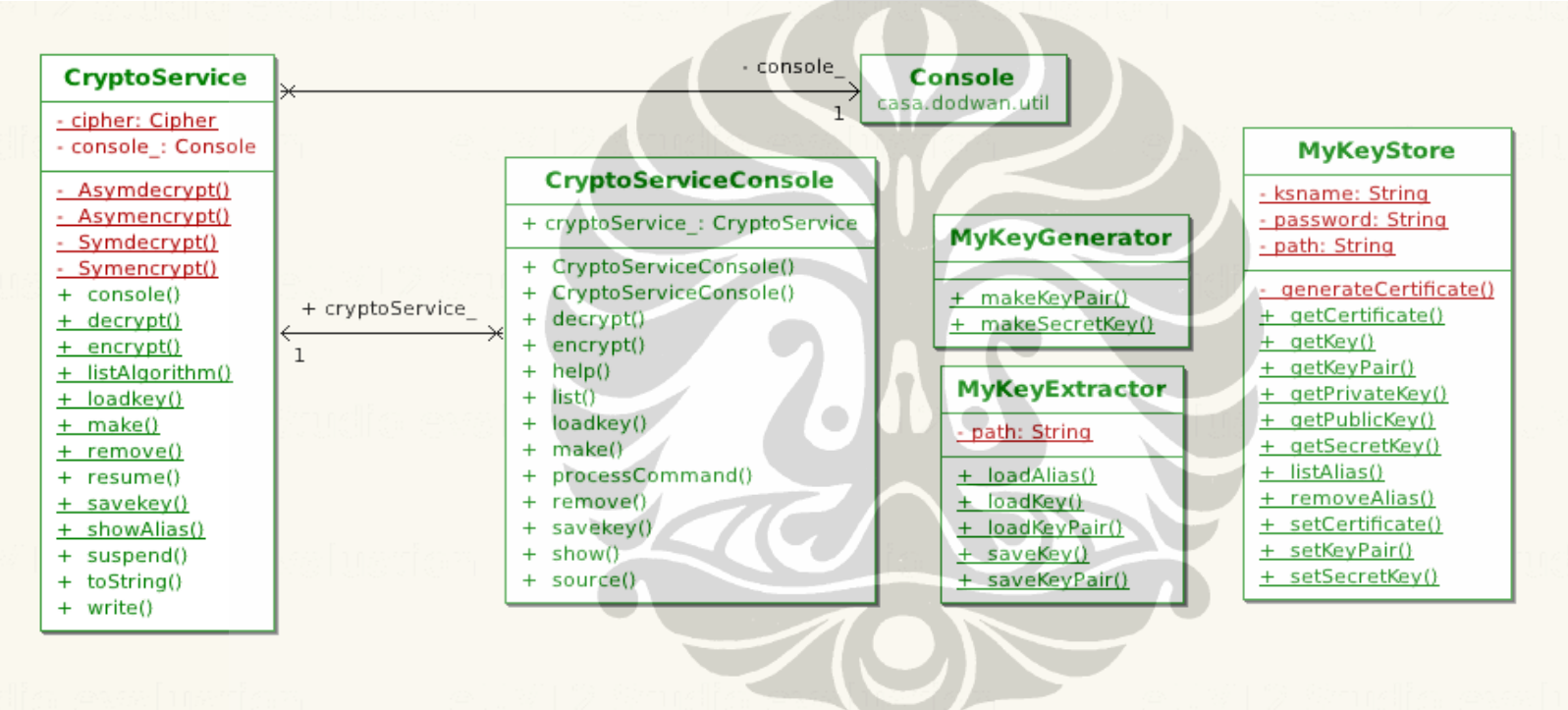


Figure 6. Class Diagram of crypto package

Note : the full documentation about crypto package, including detail of class and method, provided in section B of the appendix.

4. Conclusion

4.1. Conclusion of the Project

The conclusion of this project are :

1. Because DoDWAN was developed using JAVA language programming, that one of its feature is portability, we have the possibility of porting DoDWAN in Windows system. This porting process is successfully done without any significant problem.
2. We have successfully implements the cryptographic functionality in the DoDWAN crypto package, using the JAVA security class (JCA and JCE). This functionality can later be used to add security features of the DoDWAN.

4.2. Perspective

Here are some perspectives for future development of the DoDWAN middle ware platform :

1. We can exploits the portability feature of JAVA by porting this DoDWAN middle ware platform in other portable devices such as PDA or smart phone. Today smart phones are equipped with Wi-Fi devices, therefore to port DoDWAN in this platform, for example : Android, will be interesting. The problem is that Android using DalvikVM virtual machine. We must search the possibility to implement Sun's JavaVM in the Google's DalvikVM. This would make it possible to run JVM based applications out-of-the-box on the Android system.

2. Although we have successfully implements both symmetric and asymmetric cryptography, there is a limitation in asymmetric cryptography. The only asymmetric algorithm using cipher that supported by JCA and JCE is RSA and DSA. In this internship we just need implement the encryption process, so we don't need the DSA algorithm.

Based on book [6], section 5.4.5 RSA Encryption Limitations, “Java Cryptography Extension Practical Guide For Programmer”, 2004, page 145, in the real world, the encryption capabilities of RSA are rarely used for one simple reason: the length of the plain text that can be encrypted is limited to the size of n. In fact, the real length is even smaller than n because of the overhead introduced by the algorithms. As a result, the predominate approach is to generate a random secret key and encrypt that key with the RSA keys. The message is then encrypted using a symmetric cipher with the generated secret key.

3. In this internship we only use the standard JAVA security package provide by the SUN (JCA and JCE). There is third party library for JAVA security which provide more complete cryptographic functionality. This library call Bouncy Castle, it is free and open source. Bouncy Castle is a collection of APIs used in cryptography. It includes APIs for both the Java and the C# programming languages. According to their website, this library implements all JCA and JCE architecture and also add other functionality. With this library we can add more secure feature in DoDWAN platform.

4. According Wikipedia (http://en.wikipedia.org/wiki/Key-agreement_protocol), in cryptography, a key-agreement protocol is a protocol whereby two or more parties can agree on a key in such a way that both influence the outcome. If properly done, this precludes undesired third-parties from forcing a key choice on the agreeing parties. Protocols that are useful in practice also do not reveal to any eavesdropping party what key has been agreed upon. We can eliminate the use of keystore if we can implements this protocol, because we don't need to save the key.

Bibliography

[1] Julien Haillot, Frédéric Guidec. A Protocol for Content-Based Communication in Disconnected Mobile Ad Hoc Networks. Journal of Mobile Information Systems, 6(2):123-154, 2010.

[2] Walid Gédéon. OSGi and Apache Felix 3.0 Beginner's Guide, Packt Publishing, 2010.

[3] Julien Haillot, Frédéric Guidec. A Protocol for Content-Based Communication in Disconnected Mobile Ad Hoc Networks. In IEEE 22nd International Conference on Advanced Information Networking and Applications (AINA'08), pp. 188-195, Okinawa, Japan, March 2008.

[4] Frédéric Guidec, Yves Mahéo. Opportunistic Content-Based Dissemination in Disconnected Mobile Ad Hoc Networks. In International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM 2007), pp. 49-54, Papeete, French Polynesia (Tahiti), November 2007.

[5] David Hook. Beginning Cryptography in Java, Wrox Press, 2005.

[6] Jason Weiss. Java Cryptography Extension Practical Guide For Programmer, Morgan Kaufmann Publisher, 2004.

[7] Li Gong, Gary Ellison, Mary Dageforde. Inside Java™ 2 Platform Security: Architecture, API Design, and Implementation, Second Edition, Addison Wesley, 2003.

[8] A. Carzaniga and A. L. Wolf. Content-based Networking: a New Communication Infrastructure. In Proceedings of the NSF Workshop on an Infrastructure for Mobile and Wireless Systems, number 2538 in LNCS, pages 59--68. Springer, Oct. 2001.

[9] Jonathan B. Knudsen. Java Cryptography, May 1998.

Appendix

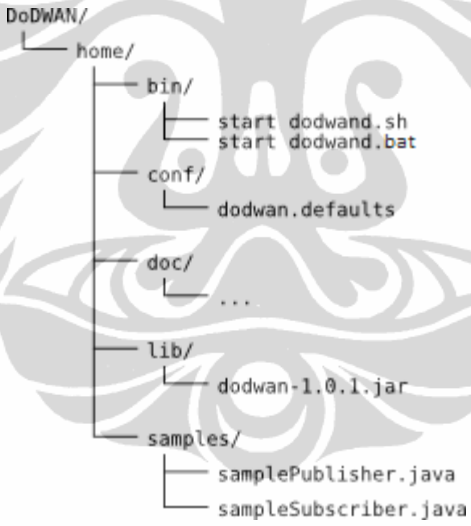
A. DoDWAN Tutorial For Windows

Introduction

This tutorial explains how to install the DoDWAN software, gives hints for configuring a network suitable for a DoDWAN application, and describes how DoDWAN can be used, first through its daemon and then through its Java API, in Windows environment (this tutorial was developed using Microsoft Windows 7).

Package Installation

After having downloaded the latest DoDWAN distribution for Windows from the CASA web page, simply unzip the package file dodwan-1.0.1-1.zip in the directory of your choice, for example, in your home directory. The package contains the following tree:



Network Configuration

Running DoDWAN in a wired LAN : the easy way

Note for Windows :

Both IPv4 and IPv6 are installed and enabled by default in most Windows Operating System (Windows XP had an optional IPv6 stack, but Windows XP SP2, Windows Vista and Windows 7 are Ipv6 enabled). You can check Ipv6 address of your network interfaces using `netsh` command in command prompt.

```
PS C:\Windows\system32> netsh interface ipv6 show address
...
Interface 23: Wireless Network Connection 2

Addr Type  DAD State   Valid Life Pref. Life Address
---------             -----------       ----------       ----------
------------------------
Other                 Deprecated          infinite        infinite
fe80::3817:4fe4:48d1:ff9c%23
...
```

Ipv6 multicast address also enabled by default in Windows. If you want to check which multicast address groups that your system had joined, use the following command :

```
PS C:\Windows\system32> netsh interface ipv6 show joins
...
Interface 23: Wireless Network Connection 2

Scope       References  Last  Address
----------  ----------  ----  --------------------------------
0                    0  Yes   ff01::1
0                    0  Yes   ff02::1
0                    1  Yes   ff02::1:ffd1:ff9c
...
```

Running DoDWAN in a wireless ad hoc network in Windows system

If you want to create wireless ad hoc network in Windows 7 using GUI follow this tutorial provided by Microsoft in this link http://Windows.microsoft.com/en-US/Windows7/Set-up-a-computer-to-computer-ad-hoc-network. You can also configure wireless settings using commands in the *netsh wlan* context of the Netsh command-line tool, which enables you to create scripts that connect to wireless ad hoc network. To list available wireless networks, run the following command :

```
PS C:\Windows\system32> Netsh wlan show networks

Interface name : Wireless Network Connection 2
There are 2 networks currently visible.

SSID 1 : eduroam
    Network type            : Infrastructure
    Authentication          : WPA2-Enterprise
    Encryption              : CCMP

SSID 2 : dodwan
    Network type            : Adhoc
    Authentication          : Open
    Encryption              : None
```

From the information above we have wireless network interface named **Wireless**

```
Network Connection 2.
```

To connect to a wireless network using Netsh, first you must have a profile saved for that network. This profile can be saved in an Extensible Markup Language (XML) file format (you can look one example of XML profile in the hints and tips parts), then you can connect to a wireless ad hoc network by loading this profile file. This following example demonstrate how to configure or added XML profile file in the system.

```
PS C:\Windows\system32> netsh wlan add profile
filename=C:\Users\wireless\Documents\"profile.xml"
Interface="Wireless Ne
twork Connection 2"

Profile DODWAN-Windows7 is added on interface Wireless Network
Connection 2.
```

filename is the path of the profile.XML file in your computer and Interface is the name of the wireless interface. In the example above we add the profile DODWAN-Windows7 in the system.

After we add the profile.XML in the system we can connect to wireless ad hoc network by using the *netsh wlan connect* command and specify a wireless profile name (which must be configured or added previously).

```
PS C:\Windows\system32> netsh wlan connect DODWAN-Windows7
Connection request was completed successfully.
```

Note : you need to specify the interface name only if you have multiple wireless network adapters. You can use the command `netsh wlan disconnect` to disconnect from all wireless networks. Netsh has many other commands for configuring wireless networking, for more information, run the following command at a command prompt.

```
PS C:\Windows\system32> netsh wlan help
```

Hints and Tips :

Setting a route for multicast networking

If DoDWAN is set to use IPv6 networking (which is the default), then a route must be established to reach the multicast group through the selected interface. Use this following command for add ipv6 multicast address in Windows :

```
PS C:\Windows\system32> netsh interface ipv6 add route
ff02::236:1:2:3/128 "Wireless Network Connection 2"
```

```
Ok.
```

In the example above `ff02::236:1:2:3/128` is the ipv6 multicast address and `"Wireless Network Connection 2"` is the name of the interface.

If DoDWAN uses IPv4 networking, then a command such as that below must be used instead:

```
PS C:\Windows\system32> netsh interface ipv4 add route
236.1.2.3/32 "Wireless Network Connection 2"

Ok.
```

Special Notes for Windows

Connect Automatically

In the case of DoDWAN, sometimes you need to connect automatically in some wireless ad hoc network, for example : you want to connect to ad hoc network named “dodwan”. If you want Windows 7 connected automatically to an ad hoc network, it needs a little trick , since Windows 7 by default doesn't allow this because of the security reasons. First, create the profile file : profile.xml, and then create a batch script file in the Startup folder of Windows 7, in order that this batch file can be executed automatically each time Windows 7 start. The batch script will add profile.xml in the system and at the same time, we connected the device with a wireless ad hoc network using netsh. Here is the file example of the profile.xml:

```
<?xml version="1.0"?>
<WLANProfile
xmlns="http://www.microsoft.com/networking/WLAN/profile/v1">
      <name>DODWAN-Windows7</name>
      <SSIDConfig>
            <SSID>
                  <hex>444F4457414E2D57494E444F575337</hex>
                  <name>DODWAN-Windows7</name>
            </SSID>
            <nonBroadcast>false</nonBroadcast>
      </SSIDConfig>
      <connectionType>IBSS</connectionType>
      <connectionMode>manual</connectionMode>
      <autoSwitch>false</autoSwitch>
      <MSM>
            <security>
                  <authEncryption>
                        <authentication>open</authentication>
                        <encryption>none</encryption>
                        <useOneX>false</useOneX>
                  </authEncryption>
            </security>
```

```
        </MSM>
    </WLANProfile>
```

And here is the batch script, and put this script in the startup folder :

```
@echo off
netsh wlan add profile filename=C:\stage\profile.xml
      interface="Wireless Network Connection 2"
netsh wlan connect ssid=DODWAN-Windows7 name=DODWAN-Windows7
interface="Wireless Network Connection 2"
```

In the example above, we create the ad hoc network with the SSID name "DODWAN-Windows7" in the profile.xml file, and then connect to this SSID using the batch script.

Wireless Ad Hoc Channel Configuration in Windows

Although the netsh command has many capabilities, but unfortunately, until this tutorial is created, we can not configure the specific channel number for our ad hoc wireless networks in Windows. For operates, dodwan needs specific channel number of the ad hoc network, for example 3. This functional depends entirely on the drivers supplied by vendors of our wireless network adapters. If the driver provides features to configure the channel number then we can apply it on dodwan. But if the driver do not provide this feature, here are some tricks that can be done.

The solution we used for this problem is to change the driver of the wireless device to use the specific channel. For example, wireless network interface that we used in this research is D-Link DWL-G122 and it has the channel number 1 by default. Windows 7 will automatically know this device and uses the generic driver netr7364.inf for this device. To force this device to use channel 3, we changed the driver netr7364.inf and changes the channel number 1 become 3.

```
...
;
; Parameters
;
RadioOnOff                  = "Radio On/Off"
Disable                     = "Disable"
Enable                      = "Enable"
IEEE802_11h                 = "IEEE802.11h"
SmartScan                   = "SmartScan"
CarrierDetect               = "Carrier Detect"

CHANNEL                     = "3"
WPS_DEVNAME_DEF_STR         = "Ralink Client"
```

If we use another wireless device, the steps to configure the number of channel is substantially the same. First, we look for the driver for this device, and then changed the driver *. inf file and change the channel number by default.

Using DoDWAN as a Daemon

Details on the DoDWAN Daemon Usage

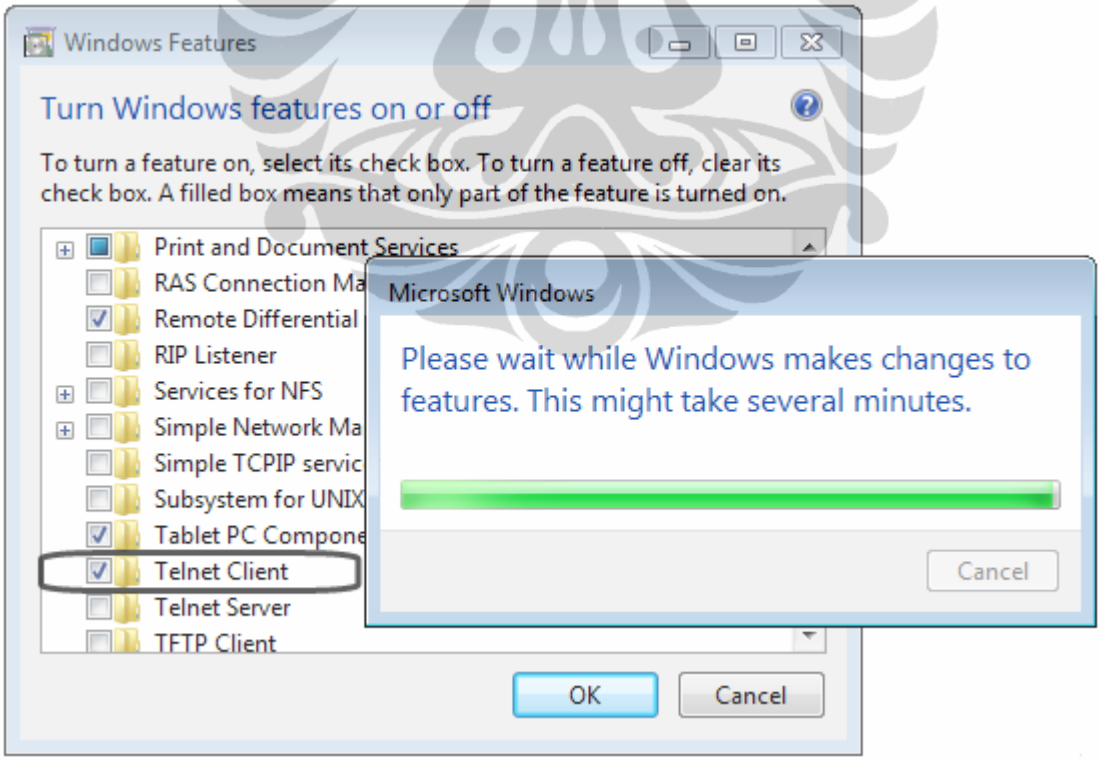
Note for Windows :
Here is the script start_dodwan. bat to replace start_dodwan.sh.

```
@echo off
java -cp %userprofile%\DoDWAN\home;%userprofile
%\DoDWAN\home\lib\dodwan-1.0.1.jar casa.dodwan.run.dodwand
```

For run the dodwan console we need the telnet program. In Windows 7, Microsoft telnet is not activated by default. If you want to activate, uses this steps :

How to Enable Default Telnet Client on Windows 7:

1. Go To Start Menu and select Control Panel
2. Click on Programs in control panel
3. Now click on **"Turn Windows Features On or Off"** under Programs and Features
4. Scroll down in displayed list and check Telnet Client and click on OK



Or you can download third party software for telnet, for example : putty.

Using the DoDWAN API

Note for Windows :
Rather than continuously compile and execute samplePublisher.java and sampleSubscriber.java files repeatedly in the command prompt, we can create a simple batch script to run this DoDWAN API sample. Create this batch script file in the samples folder.

```
REM script publisher
@ECHO OFF
javac -cp %userprofile%\DoDWAN\home\lib\dodwan-1.0.1.jar
samplePublisher.java
java -cp %userprofile%\DoDWAN\home;%userprofile
%\DoDWAN\home\lib\dodwan-1.0.1.jar samples.samplePublisher
```

```
REM script publisher
@ECHO OFF
javac -cp %userprofile%\DoDWAN\home\lib\dodwan-1.0.1.jar
sampleSubscriber.java
java -cp %userprofile%\DoDWAN\home;%userprofile
%\DoDWAN\home\lib\dodwan-1.0.1.jar samples.sampleSubscriber
```

B. Documentation Of The Crypto Package

Package casa.dodwan.crypto

Class Summary	
CryptoService	Class for giving the cryptography service in DoDWAN
CryptoServiceConsole	Class for giving the cryptography service in the DoDWAN console
MyKeyExtractor	Class for Extracting Key from and to file
MyKeyGenerator	Class for generating Key (Symetric and Asymetric)
MyKeyStore	Class for Key Management in the Java KeyStore

Class MyKeyGenerator

Method Summary	
Static java.security.KeyPair	**makeKeyPair**(int size, java.lang.String algo) Method for generate KeyPair
Static javax.crypto.SecretKey	**makeSecretKey**(int size,

	java.lang.String algo) Method for generates SecretKey

Class MyKeyExtractor

Field Summary	
private static java.lang.String	**path** Key file path

Method Summary	
Static java.lang.String	**loadAlias**(java.lang.String filename) Method for loads alias from file
Static java.security.Key	**loadKey**(java.lang.String filename) Method for loads key from the file
Static java.security.KeyPair	**loadKeyPair**(java.lang.String filename) Method for loads keypair from file
Static boolean	**saveKey**(java.lang.String alias, java.security.Key key, java.lang.String filename) Method for saves key in to file
Static boolean	**saveKeyPair**(java.lang.String alias, java.security.PrivateKey privk, java.security.PublicKey pubk, java.lang.String filename) Method for saves keypair in to file

Class MyKeyStore

Field Summary	
private static java.lang.String	**ksname** KeyStore filename
private static java.lang.String	**password** password for the keystore
private static java.lang.String	**path** Keystore path file

Method Summary	
private static java.security.cert.X509Certificate	**generateCertificate**(java.lang.String dn, java.security.KeyPair pair, int days, java.lang.String algorithm) Create a self-signed X.509 Certificate
Static java.security.cert.Certificate	**getCertificate**(java.lang.String alias) Method for gets certificate
Static java.security.Key	**getKey**(java.lang.String alias) Method for gets key from the keystore
Static java.security.KeyPair	**getKeyPair**(java.lang.String alias) Method for retrieves KeyPair from the keystore
Static java.security.PrivateKey	**getPrivateKey**(java.lang.String alias) Method for retrieves PrivateKey from the keystore
Static java.security.PublicKey	**getPublicKey**(java.lang.String alias) Method for retrieves PublicKey from the certificate in the keystore
Static javax.crypto.SecretKey	**getSecretKey**(java.lang.String alias) Method for retrieves Secret Key from the keystore
Static java.lang.String	**listAlias**() Method for list all alias in the key store
Static boolean	**removeAlias**(java.lang.String alias) Method for delete/remove alias from the Keystore.
Static void	**setCertificate**(java.lang.String alias, java.security.cert.Certificate cert) Method for saves certificate (Public Key) to the keystore there is public key inside the certificate
Static void	**setKeyPair**(java.lang.String alias, java.security.KeyPair kp) Method for saves KeyPair (Private and Public Key) to the keystore
Static void	**setSecretKey**(java.lang.String alias, javax.crypto.SecretKey sk) Method for saves the secret key to the keystore

Class CryptoService

Field Summary	
private static javax.crypto.Cipher	**cipher** Cipher initialization
private casa.dodwan.util.Console	**console_**

Method Summary	
private static java.lang.String	**Asymdecrypt**(java.lang.String alias, java.lang.String cipherText) Method for message decryption using Asymmetric Cryptography Message is encrypted using Private Key
private static java.lang.String	**Asymencrypt**(java.lang.String alias, java.lang.String message) Method for message encryption using Asymmetric Cryptography Message is encrypted using Public Key
casa.dodwan.util.Console	**console**()
Static java.lang.String	**decrypt**(java.lang.String alias, java.lang.String cipherText) Method for messages decryption Based on type of the key owned by alias, if the key is secret key, than do the symmetric decryption else, do asymmetric decryption
Static java.lang.String	**encrypt**(java.lang.String alias, java.lang.String message) Method for messages encryption Based on type of the key owned by alias, if the key is secret key, than do the symmetric encryption else, do asymmetric encryption
Static java.lang.String	**listAlgorithm**() Method for listing all supported algorithm
Static boolean	**loadkey**(java.lang.String filename) Method for loads key from files and save in the keystore
Static boolean	**make**(java.lang.String alias, int size, java.lang.String algo) Method for creates key and save in the keystore

Method Summary	
Static boolean	**remove**(java.lang.String alias) Method for delete/remove alias from the keystore
void	**resume**() Resume.
Static boolean	**savekey**(java.lang.String alias, java.lang.String filename) Method for get key from the key store and save in file
Static java.lang.String	**showAlias**() Method for show all alias in the keystore
void	**suspend**() Suspend.
private static java.lang.String	**Symdecrypt**(java.lang.String alias, java.lang.String cipherText) Method for message decryption using Symmetric Cryptography Message is encrypted using Secret Key
private static java.lang.String	**Symencrypt**(java.lang.String alias, java.lang.String message) Method for message encryption using Symmetric Cryptography Message is encrypted using Secret Key
java.lang.String	**toString**()
void	**write**(casa.dodwan.message.Message doc) Writes an object of type E to the sink.

Class CryptoServiceConsole

Field Summary	
CryptoService	**cryptoService_**

Constructor Summary	
CryptoServiceConsole(CryptoService cryptoService)	
CryptoServiceConsole(java.lang.String prompt, CryptoService cryptoService)	

Method Summary	
void	**decrypt**(java.lang.String[] arguments, java.io.PrintStream out)

Method Summary	
void	**encrypt**(java.lang.String[] arguments, java.io.PrintStream out)
void	**help**(java.lang.String offset, java.io.PrintStream out)
void	**list**(java.lang.String[] arguments, java.io.PrintStream out)
void	**loadkey**(java.lang.String[] arguments, java.io.PrintStream out)
void	**make**(java.lang.String[] arguments, java.io.PrintStream out)
void	**processCommand**(java.lang.String[] arguments, java.io.BufferedReader reader, java.io.PrintStream out)
void	**remove**(java.lang.String[] arguments, java.io.PrintStream out)
void	**savekey**(java.lang.String[] arguments, java.io.PrintStream out)
void	**show**(java.lang.String[] arguments, java.io.PrintStream out)
void	**source**(java.lang.String[] arguments, java.io.PrintStream out)

C. Cryptography Console Help

Here are the cryptography console help, consists all the command line for cryptography service in DoDWAN console. After running DoDWAN daemon, try to log in DoDWAN port using telnet, with the command telnet. If DoDWAN daemon already running then we can access the DoDWAN console.

```
user@pc-mna-225:~$ telnet localhost 8500
Trying ::1...
Connected to localhost.localdomain.
Escape character is '^]'.
%
```

Here are some functionality provide by cryptography service :

1. For help use this command :

```
% d kr
[ma]ke <alias> <size> <algorithm>
[re]move <alias>
[lo]adkey <alias> <filename>
```

```
[sa]vekey <alias> <filename>
[li]stalgo
[sh]owalias
[en]crypt <alias> <message>
[de]crypt <alias> <encoded message>
```

2. For list all supported algorithm used for encryption/decryption process :

```
% d kr li
Supported : RSA
Supported : DES
Supported : DESede
Supported : DESedeWrap
Supported : PBEWithMD5AndDES
Supported : PBEWithMD5AndTripleDES
Supported : PBEWithSHA1AndRC2_40
Supported : PBEWithSHA1AndDESede
Supported : Blowfish
Supported : AES
Supported : AESWrap
Supported : RC2
Supported : ARCFOUR
```

3. For show all alias in the keystore :

```
% d kr sh
alias : charlie (key pair)
alias : alice (key pair)
alias : bob (public key)
```

4. For generate key :

```
% d kr ma andysk 56 DES
Key saved succesfully in Keystore.
% d kr ma andykp 512 RSA
Key saved succesfully in Keystore.
```

5. For load key from the keystore, then save in a file :

```
% d kr sa andykp keypair
Key saved succesfully in file : keypair
```

6. For delete alias from the keystore :

```
% d kr re andykp
Alias : andykp has been removed from keystore.
```

7. For load key from file, then save in keystore :

```
% d kr lo keypair
Key loaded succesfully, and saved in Keystore.
```

8. For encrypted/decrypted some message using symmetric key :

```
% d kr en andysk "mon message"
ZkhHxnTFQ/Nvt3KTPSvyyA==
% d kr de andysk "ZkhHxnTFQ/Nvt3KTPSvyyA=="
mon message
```

9. For encrypted/decrypted some message using asymmetric key :

```
% d kr en andykp "My Message"
e40sYJ35Hnuoec3a+FHo0k/qmkdkUPu3UApGygPMa9j1O1oEekmoyXxOinuy
fIeNIFwfqQDEx5JtQBb6m2/8uQ==
% d kr de andykp
"e40sYJ35Hnuoec3a+FHo0k/qmkdkUPu3UApGygPMa9j1O1oEekmoyXxOinu
yfIeNIFwfqQDEx5JtQBb6m2/8uQ=="
My Message
```