# Implementation and Performance Evaluation of XTR over Wireless Network

By

Basem Shihada

bshihada@bbcr.uwaterloo.ca

Dept. of Computer Science
200 University Avenue West
Waterloo, Ontario, Canada
(519) 888-4567 ext. 6238

CS 887 Final Project

19th of April 2002

**Implementation and Performance Evaluation of XTR over Wireless Network**

## 1. Abstract

*Wireless systems require reliable data transmission, large bandwidth and maximum data security. Most current implementations of wireless security algorithms perform lots of operations on the wireless device. This result in a large number of computation overhead, thus reducing the device performance. Furthermore, many current implementations do not provide a fast level of security measures such as client authentication, authorization, data validation and data encryption.*

*XTR is an abbreviation of Efficient and Compact Subgroup Trace Representation (ECSTR). Developed by Arjen Lenstra & Eric Verheul and considered a new public key cryptographic security system that merges high level of security GF($p^6$) with less number of computation GF($p^2$). The claim here is that XTR has less communication requirements, and significant computation advantages, which indicate that XTR is suitable for the small computing devices such as, wireless devices, wireless internet, and general wireless applications. The hoping result is a more flexible and powerful secure wireless network that can be easily used for application deployment.*

*This project presents an implementation and performance evaluation to XTR public key cryptographic system over wireless network. The goal of this project is to develop an efficient and portable secure wireless network, which perform a variety of wireless applications in a secure manner. The project literately surveys XTR mathematical and theoretical background as well as system implementation and deployment over wireless network. The utilization of secure wireless network accomplishes several enhancements such as better wireless CPU utilization, distribution of light weight security process, adjustment of the wireless device bandwidth and enabling less number of data communications. Performance results from experiments on the test bed validate many of the advantages of using this public key system over wireless networks.*

## 2. Introduction

XTR is a public key cryptographic system, so it is similar to Rivest, Shamir and Adleman (RSA) and Elliptic Curve Cryptosystems (ECC) in which part of the cryptographic key made public, or transferred via unsecured connections. RSA security depends on the difficulty of factoring large numbers in a finite field defined by modular arithmetic, where the modulus is prime integer. This factoring problem called Discrete Logarithm problem (DL). ECC security depends similarly on factoring large numbers, but the field is generated not by primitive roots of prime numbers, but by points on a curve that cross-exact integer coordinates. XTR is a cryptosystem based on multiplicative groups of finite fields.

As proposed in the abstract, XTR has less communication requirements, and significant computation advantages, which leads to a conclusion that XTR could be suitable for the

small computing devices such as, wireless devices, wireless internet, and general wireless applications. Some candidate applications to this system are WAP, SSL, and Smartcards. The following section focuses on wireless security in view of historical and technical aspects.

### 2.1 Wireless security

Wireless information system security is categorized into four areas, data security, computer security, network security, and wireless channel security[1,2].

In free-space transmission of wireless network, when one sends a message over the radio path, everyone equipped with a suitable transceiver within the range of the transmission can eavesdrop on the message. The sender or intended receiver has no means of knowing if the transmission has been intercepted or not, so the eavesdropping is virtually undetectable. The *Wall Street Journal* (April 27, 2001) described two hackers with a laptop and a boom antenna driving around Silicon Valley listening to network after network.

The wireless network, as part of the enterprise network, offers one interface to the attacker without requiring any physical arrangements such as network cables. The mobile computing devices are easy to loss and consequently can be misused by malicious attackers.

Russell [3] also lists some other obvious and probable security issues for wireless networks: unintentional interference, user location, jamming, and service degradation. All these require network security technologies be used to get better wireless network security. Good wireless network security solutions should also consider the characteristics of wireless networks, such as limited computing power and the efficient use of mobile device's power source. XTR seems to provide some of these requirements.

According to [4] Virtual Private Network (VPN) is a private network running over a shared network infrastructure (Internet) by creating tunnels to provide secure interconnection for corporate networks and remote users. Data packets travel through the private tunnel that simulates a "peer-to-peer" connection, which means access is controlled to permit peer connections only within a defined community of interest, and is constructed though a common underlying non-exclusive communication medium. Any VPN implementation allows for the opaque transport of frames as packet payloads across an IP network.

Adapting the concept of VPN, XTR is the selected as security solution. XTR can be described as a set of security applications that includes Diffie-Hellman key agreement (XTR-DH) and ElGamal encryption and decryption applications (XTR-ElGamal).

## 2.2 Cryptography

Cryptography is a field of study which provides the required techniques, algorithms, and implementations of data confidentiality. There are several subfields within cryptography such as, authentication, authorization and encryption. Authentication create a verify messages digests and digital signatures. These algorithms can ensure that any data is authentic; that it originates from the person who claims to have originated it and has not been accidentally or maliciously modified in communication. Authorization is to ensure that the service is accessed by the user who is only allowed to access it. Data encryption is a set of algorithms and functions used to cipher a plain text; these encryption functions must be strong enough so no third party can retrieve the original data.

Within cryptography we define XTR as a public key cryptosystem. Public key cryptosystems were invented in the late 1970's, with some help from the development of complexity theory [5]. It was observed that based on a problem so difficult that it would need thousands of years to solve, and with some luck, a cryptosystem could be developed which would have two keys, a secret key and a public key. With the public key one could encrypt messages, and decrypt them with the private key. Thus the owner of the private key would be the only one who could decrypt the messages, but anyone knowing the public key could send them in privacy [5,6].

Within public key cryptosystem there has been a new mechanism to address the steps of creating the secret key. In other words, it is defined a key exchange algorithm. Diffie and Hellman used ideas from number theory to construct a key exchange protocol that started the period of public key cryptosystems [5]. Shortly after that Rivest, Shamir and Adleman developed a cryptosystem that was the first real public key cryptosystem capable of encryption and digital signatures [5]. Later several public cryptosystems followed using many different underlying ideas. Many of them were soon proven to be insecure. However, the Diffie-Hellman protocol and RSA appear to have remained two of the strongest up to now [6]. A version of Diffie-Hellman key exchange has been implemented over XTR public key cryptosystem.

## 2.3 Paper outline

Section 3 surveys the existing public key cryptosystems, with a summary of the basic characteristics of each system. It also illustrates the mathematical idea behind each. Furthermore, a comparison study between XTR, ECC, and RSA is provided. Section 4 focuses on the XTR mathematical background and algorithms. Within this section Galois fields, XTR subgroup & super group, and element trace is discussed. Section 5 specifically illustrates the XTR system construction. Section 6 shows an implementation demonstration and illustrates the steps of running this system. Snapshots of some pre-runes operations have been provided. Section 7 describes the experimental results of the system overall performance, the performance reflects the levels of analysis that shows the total operation time, encryption, decryption time, and the wireless device CPU/Memory usability during different stages of the system. Section 8 summarizes the achieved goals of this approach and some suggestions for future work.

## 3 Public-key Cryptographic Systems

### 3.1 Diffie-Hellman

Is a commonly used protocol for key exchange In many cryptographical protocols two parties wish to begin communicating. However, assume they do not initially possess any common secret and thus cannot use secret key cryptosystems. The key exchange by Diffie-Hellman protocol remedies this situation by allowing the construction of a common secret key over an insecure communication channel. It is based on a problem related to discrete logarithms, namely the Diffie-Hellman problem. This problem is considered hard, and it is in some instances as hard as the discrete logarithm problem. Using a large period generator is important to consider the generated group secure [5,6,7].

Algorithm:
Assume that Alice wants to exchange a secrete key with Bob. First, Bob and Alice agreed on a certain public values p and g.

1. Alice picks a secret value $\alpha \in Z$ at random. Similarly, time Bob picks a secret value $\beta \in Z$.
2. Alice computes $T\alpha = g^{\alpha} \pmod p$. Bob computes $T\beta = g^{\beta} \pmod p$
3. Alice sends $T\alpha$ to Bob and Bob sends $T\beta$ to Alice.
4. Alice computes $(T\beta)^{\alpha} \pmod p$ and Bob computes $(T\alpha)^{\beta} \pmod p$
5. The resultant value K is the shared secret key.

### 3.2 Digital Signature

Integrity, authentication, non-repudiation and certification is achieved by some versions of digital signatures. The underlying algorithm DSA (Digital Signature Algorithm) is similar to the one used by ElGamal or by the Schnorr signature algorithm. Also it is fairly efficient, although not as efficient as RSA for signature verification. RSA is a public key encryption technique named after its inventers: RSA. It's commonly used with key lengths of 512 bits and based on modular arithmetic [5,6,7].

Algorithm:
The basic steps to construct the RSA algorithm are:

1. Choose two random large prime numbers p, q (typically 265-bits each)
2. Define $n = p \cdot q$
3. Choose a number e s.t. $e < n$ and e is relatively prime to $(p-1)(q-1)$
4. Find the number d s.t. $d \cdot e \bmod (p-1)(q-1) = 1$
5. Public key is (e,n) and private key is (d,n)
6. Encryption is defined as cipher = $(\text{message})^{e} \bmod n$
7. Decryption is defined as m = $(\text{cipher})^{d} \bmod n$

### 3.3 ElGamal

ElGamal public key cryptosystem is an extension of Diffie-Hellman's original idea on generating shared secret key. Essentially, it generates a shared secret and uses it as a one-time pad to encrypt one block of data. The Generalized ElGamal public key cryptosystem over finite field is defined as follows [5,6].

Definition [5,6]:
Let G be a fine group with group operation o, and let $\alpha \in G$ be an element such that the discrete log problem in H is intractable, where $H = \{\alpha^i : i \geq 0\}$ is the subgroup generated by $\alpha$. Let P = G, C = G X G, and define
$K = \{(G, \alpha, a, \beta) : \beta = \alpha^a\}$.
The values $\alpha$ and $\beta$ are public, and a is secret
For $K = (G, \alpha, a, \beta)$, and for a (secret) random number $k \in Z$, define
$E_k(x,k) = (y_1, y_2)$, where $y_1 = a^k$ and $y_2 = x \text{ o } \beta^k$
For a cipher text $y = (y_1, y_2)$, define
$D_k(y) = y_2 \text{ o } (y_1^a)^{-1}$

### 3.4 Elliptic cure cryptosystem

Elliptic curve cryptosystems are another way of implementing discrete logarithm methods. An elliptic curve is defined as a set of points that satisfy the equation y2 = x3 + ax + b when considered in finite field of characteristic p, where p> 3. A slightly different equation is needed for the cases of small characteristic, p = 2 and p = 3. The points on elliptic curve can be added together and then form a structure called a group. To mathematically define the concept of elliptic curves we define the following

Definition:
Let p > 3 be prime. The elliptic curve $y^2 = X^3 + a X + b$ over $Z_p$ is the set of solutions $(x,y) \in Z_p \text{ X } Z_p$ to the congruence $y^2 = X^3 + a X + b \pmod p$
Where a, b $\in$ Z are constants such that $4a^3 + 27b^2 \neq 0 \pmod p$, together with a special point O called the point at infinity [5].

Elliptic curves is adapted efficiently in hardware and software, and they compete well in speed with cryptosystems such as RSA and DSS. There are several standardization attempts for elliptic curve cryptosystems, for example, ECDSA by ANSI [7].

XTR algorithm becomes a competitor for elliptic curves. However, elliptic curves appear to be better in level of security, but not in level of computations [10].

### 3.5 XTR

XTR is a public key cryptosystem developed by Arjen Lenstra and Eric Verheul. XTR uses a specific multiplicative group of a particular finite field $GF(p^6)$ as its underlying group [8,9]. However, XTR has novel features such as needing only 1/3 of the bits for signatures and encrypte messages. This is achieved using a specific trace-representation

of the elements of this group, and performing all computations using this representation. All discrete logarithm based public key algorithms can be implemented with XTR ideas [8,9].

According to Lenstra and Verheul the algorithm is efficient and might be a good substitute to elliptic curves, DSS and even RSA. It has the advantage over elliptic curves that it is based essentially on the same discrete log problem as, say, DSS, which may help cryptographers and others to accept it as a faster and strong algorithm [7,10].

### 3.6 Cryptographic system comparison

Based on a comparison study of public key cryptosystem proposed by [9,11], using Pentium III 450 MHz and 96 MB of Ram the following table shows a performance results of XTR, ECC, and RSA.

| | XTR | ECC | RSA |
|---|---|---|---|
| Operation Speed | Fastest 23ms | Fast 28ms | Slow 5ms for 32-bit |
| Key Length | Short 170-bit | Short 170-bit | Long 1020-bit |
| Key Selection | Simplest | Hard | Simple |

XTR shows lots of improvements in terms of operation speed, key length and key selection. The following sections focus on XTR public-key cryptographic system. They provide a literature survey of XTR, based on the mathematical and theoretical backgrounds as well as a system implementation and demonstration.

## 4 XTR Overview

### 4.1 Discrete logarithm

XTR is a discrete logarithmic problem. “Discrete logarithm is defined as the problem of finding n given only some y such that $y = g^n$. The problem is easy for integers, but when working with different setting it becomes un-traceable [5,6,12]. For example, working with fields such as, Galois Field. When the chosen integer m is equal p such that p a prime number, is often called a prime field or a Galois field, GF(p). The discrete logarithm problem in the finite field GF(p) is then defined by given two positive non-zero integers a , g (both less than p), compute n such that $a = g^n \pmod p$. We can choose g so that a solution for n exists for any non-zero a. To make this problem cryptographically hard p should be a large prime number (about 10300 and n, in general, of same magnitude)” [5,6,12].

Definition [5,6]:
Discrete Log problem is to fine an integer such that $\alpha a = \beta \pmod n$. Since gcd ($\alpha$, n) = 1, $\alpha$ has a multiplicative inverse modulo n, and we can compute $\alpha^{-1} \bmod n$ easily using Euclidean algorithm. Then we can solve for a, obtaining $\log_\alpha \beta = \beta \alpha^{-1} \bmod n$.

Within this new shape the problem now considered as hard factoring. There is little hope to find a polynomial time algorithm for the computation of discrete logarithms in GF(p). In such a case it would be likely that factoring problems also could be efficiently solved.
The discrete logarithm problem is applied on XTR public key cryptosystem. The reason for this is that there are many methods for computing discrete logarithms over XTR but it appears even harder to solve the discrete over $GF(p^6)$. This has also the effect that there are some key size benefits for using XTR based public key cryptosystems as opposed to factoring based cryptosystems [7,12].

### 4.2 XTR uses Galois Fields

Galois Filed is an example of finite filed of a q element such that $q = p^n$ where p is a prime and $n > 1 \in Z$. A Galois Field is a finite extension field of characteristic p and extension of degree n [6,7].

Definition [5,6]:
Suppose p is prime. Define $Z_p[x]$ to be the set of all polynomials in the indeterminate x. by defining addition and multiplication of polynomials in the usual way (and reducing coefficients modulo p), we construct a ring. For $f(x), g(x) \in Z_p[x]$, we say that f(x) divides g(x) (notation $f(x) \mid g(x)$ ) if there exist $q(x) \in Z_p[x]$ such that $g(x) = q(x) \cdot f(x)$

XTR uses Galois Fields $GF(p^6)$. For example, GF(11) = {0,1,2,…,10} define 2 as a primitive element of GF(11), then the field elements will be $2^0 = 1 \bmod 11$, $2^1 = 2 \bmod 11$, …, $2^4 = 5 \bmod 11$, $2^5 = 10 \bmod 11$, ….., $2^{10} = 1 \bmod 11$.

### 4.3 XTR super group $GF(p^6)$ and sub group $GF(p^2)$

XTR uses a base generator of full-multiplicative group of a finite filed with specific changes. It replaces this generator by the generator of a relatively small subgroup of sufficiently large prime order q. "XTR uses subgroup of prime order q of order $p^2-p+1$ subgroup of $GF(p^6)$ and the order q subgroup g generated by g is referred to as the XTR subgroup. The XTR super group is not contained in any proper subfield of $GF(p^6)$. Combined with the choice of q it follows that computing discrete logarithms in g is as hard, in general, as it is in $GF(p^6)$" [8,9].

The reason that XTR uses this specific subgroup g is not just that it provides full $GF(p^6)$ security, but also very efficient representation, at a small cost. "For example, if one is willing to give up the distinction between elements and their conjugates over $GF(p^2)$, then not only elements of the XTR super group can be represented using an element of $GF(p^6)$ as opposed to $GF(p^6)$. But also calculations take place in $GF(p^2)$ instead of $GF(p^6)$ and can thus be performed much faster than usual. In other words, working in $GF(p^2)$ means that working with a second degree polynomials but working with $GF(p^6)$ means working with degree six. Second degree polynomials are much easier to factor and reduced via the modulus, with a decrease of the computations in generating public data, key selection, encryption and decryption. While transferring the computation from $GF(p^6)$ to $GF(p^2)$ we already achieved a factor of three computation reduction" [8,9].

### 4.4 Traces

XTR uses the trace over $GF(p^2)$ to represent an element g of the order $p^2-p+1$ subgroup $GF(p^6)$ to achieve a factor 3 computation size reduction and thus faster calculations. "The conjugates over $GF(p^2)$ of an element $h \in GF(p^6)$ are h, $h^{p2}$, and $h^{p4}$. We define the trace of an element h $Tr(h) = h + h^{p2} + h^{p4}$.
From this definition and the fact that $h^{p6} = 1$ it follows that $Tr(h)^{p2} = Tr(h)$, so that $Tr(h)^2$ $GF(p^2)$.
Generally, the conjugates "of g are g, $g^{p-1}$ and $g^{-p}$, so that $Tr(g) = g + g + g^{p-1} + g^{-p}$. It follows that the product of the conjugates equals 1, so that the polynomial
$(X - g)(X - g^{p-1})(X - g^{-p})$ has the form $X^3 - Tr(g)X^2 + uX - 1$, where
$u = g \cdot g^{p-1} + g \cdot g^{-p} + g^{p-1} \cdot g^{-p} = g^p + g^{1-p} + g^{-1} = Tr(g)^p \in GF(p^2)$
(the last equality follows from $1 - p = - p^2$ and $- 1 = p^2 - p$, both modulo $p^2 - p + 1$). Thus
$(X - g)(X - g^{p-1})(X - g^{-p}) = X^3 - Tr(g)X^2 + Tr(g)^pX - 1 \in GF(p^2)[X]$ is actually the minimal polynomial of g over $GF(p^2)$, and this polynomial (and thereby g's conjugates) is fully determined by Tr(g). This is the fundamental observation underlying XTR" [8,9].

"The same holds for any power of g: for any integer n the conjugates of $g^n$ are the roots of $X^3 - Tr(g)X^2 + Tr(g)^pX - 1 \in GF(p^2)[X]$ and the latter polynomial is fully determined by $Tr(g^n)$. This observation is useful for cryptographic purposes if there is a way to efficiently compute $Tr(g^n)$ given Tr(g): in cryptographic protocols $g \in GF(p^6)$ can then be replaced by $Tr(g^n) \in GF(p^2)$ , thereby obtaining a saving of a factor 3 in the representation size.

Lemma:
i. $c = c1$.
ii. $c_{-n} = c_{np} = c^p_n$ for $n \in Z$.
iii. $c_n \in GF(p^2)$ for $n \in Z$.
iv. $c_{u+v} = c_u \cdot c_v - c_v^p \cdot c_{u-v} + c_{u-2v}$ for $u, v \in Z$.
v. Either all $h_j$ have order dividing $p^2-p+1$ and $> 3$ or all $h_j \in GF(p^2)$. In particular, F(c, X) is irreducible if and only if its roots have order dividing $p^2-p+1$ and $> 3$.
vi. F(c, X) is reducible over $GF(p^2)$ if and only if $c_{p+1} \in GF(p)$.

Given Tr(g), how to compute $Tr(g^n)$? First we define the following:
– c considered of the form Tr(g) for g of order $> 3$ and dividing $p^2-p+1$
– $c = Tr(g)$
– $c_n = Tr(g^n)$
– n is random Integer
– $S_n(c) = (c_{n-1}, c_n, c_{n+1}) \in GF(p^2)^3$ .

Algorithm 1 (Computation of $S_n(c)$ given n and c)
Input: c , n
Output: $S_n(c)$
– If $n < 0$, apply this algorithm to - n and c, and apply Lemma .ii to the resulting value.
– If $n = 0$, then $S0(c) = (c^p , 3, c)$ (cf. Lemma .ii).

– If n = 1, then S1(c) = (3, c, $c^2$- $2c^p$ )
– If n = 2, use S1(c) to compute c3 and thereby S2(n).
– Otherwise, to compute Sn(c) for n > 2 define Si(c) = $S_{2i+1}$(c) and let, m = n. If m is even, then replace m by m - 1. Let m = 2m + 1, k = 1, and compute $S_k(c) = S_3(c)$
Let m = the binary representation of the value, in succession does the following:
• If $m_j = 0$ then use
$S_k(c) = (c_{2k}, c_{2k+1}, c_{2k+2})$ to compute $S_{2k}(c) = (c_{4k}, c_{4k+1}, c_{4k+2})$

• If $m_j = 1$ then use
$S_k(c) = (c_{2k}, c_{2k+1}, c_{2k+2})$ to compute $S_{2k+1}(c) = (c_{4k+2}, c_{4k+3}, c_{4k+4})$

As a result of this algorithm we conclude that given the representation Tr(g) ∈ GF($p^6$) of the conjugates of g, the representation Tr($g^n$) ∈ GF($p^2$) of the conjugates of the $n^{th}$ power of g can be computed at the cost of 8 log 2 (n) multiplications in GF(p), for any integer n. This compares quite favorably to the speed of the computation of $g^n$ ∈ GF($p^2$) given g ∈ GF($p^6$)" [8,9].
We conclude with the following features:
$S_n$(c) Algorithm has a Complexity of $8\log_2$ n multiplication and a space: 2P, where P is the length of P. Compared with traditional subgroups system which has a complexity if 32.4 $\log_2$ n with space: 6P [8,9].

**5 XTR System Construction**

XTR cryptosystem requires three global values p, q, Tr(g). "There are some rules which define the ways of creating each of them**.** Primes p , *q* are chosen in such a way that *q* | ($p^2$-p+1) such that the resulting fields and subgroups are large enough to withstand known attacks. It's also found that it is more efficient if p = 2 mod 3. Primes p that is 1 mod 3 can be used as well, but they may not always achieve the same speed. Let P and Q denote the bit lengths of the primes p and q to be generated, Respectively" [8,9].

Algorithm 2
Finite field and subgroup size *p , q* are about 170-bits primes.
–Find random r ∈ Z s.t. q = $r^2$ + r + 1 is a Q-bit prime
–Find random k ∈ Z s.t. p = r + k . q = k $r^2$ +(1-k) r+k is a P-bit 2 mod 3 prime

Given p , *q* >3 find c ∈ GF($p^2$) s.t. c =Tr(g) for g ∈ GF($p^6$) of order q | ($p^2$-p+1) how to find Tr(g).

Algorithm 3
1. Pick c ∈ GF($p^2$)\GF(p), compute $c_{p+1}$using alg.1, s.t. n = p+1
2. If $c_{p+1}$ ∈ GF(p) then return to step 1
3. Compute $c_{(p2-p+1)/q}$ using alg.1 s.t. n = ($p^2$-p+1)/q
4. If $c_{(p2-p+1)/q}$ = 3 then return to step 1
5. Let Tr(g) = $c_{(p2-p+1)/q}$

## 6 System Demonstration

This section shows the required steps to run the XTR implementation over wireless network. Furthermore, it illustrates how it operates under different conditions, such as different traffic load and wireless distances. The implemented XTR consist of two applications written in C and uses free library for very long values called lip.h, in addition to lip.h [15], segments from free implementation of XTR provided by both Lenstra and Verheul are also used [13]. xtrServer is the server application and xtrClient is the client application. After the server launch the client connects to the server and retrieves the generated public data, then initializes the encryption and decryption processes. Since the purpose of the implantation is to generate as much performance data as possible, every operation is stored in a separate performance file.

### 6.1 Test bed Description

The test bed of this project is built on wireless LAN technology. The mobile node is an PII Celeron 900 computer running Linux 2.4.1-20 kernel. The antenna used by the AirPort Card is already built into the Ethernet. The Ethernet is an implementation of WAP provided by lucent technologies with turbo 11.0 Mb speed. The AirPort Base Station is configured as a wireless bridge although it also provides NAT and DHCP functionality [14]. Wireless Encryption Protocol (WEP) is disabled because XTR is used to secure the wireless communication. The AirPort Base Station is connected directly to the network of the Faculty of Computer Science at Dalhousie University, which in this case is the "public network". The XTR server is running on a PC running Linux as stated before. On the other hand, the client is running on a different system running Linux Depian. So the host-to-host XTR connection is created between the mobile node and the client system. Figure 1 illustrates the test bed topology. Figure 2 shows a picture of the used air port.

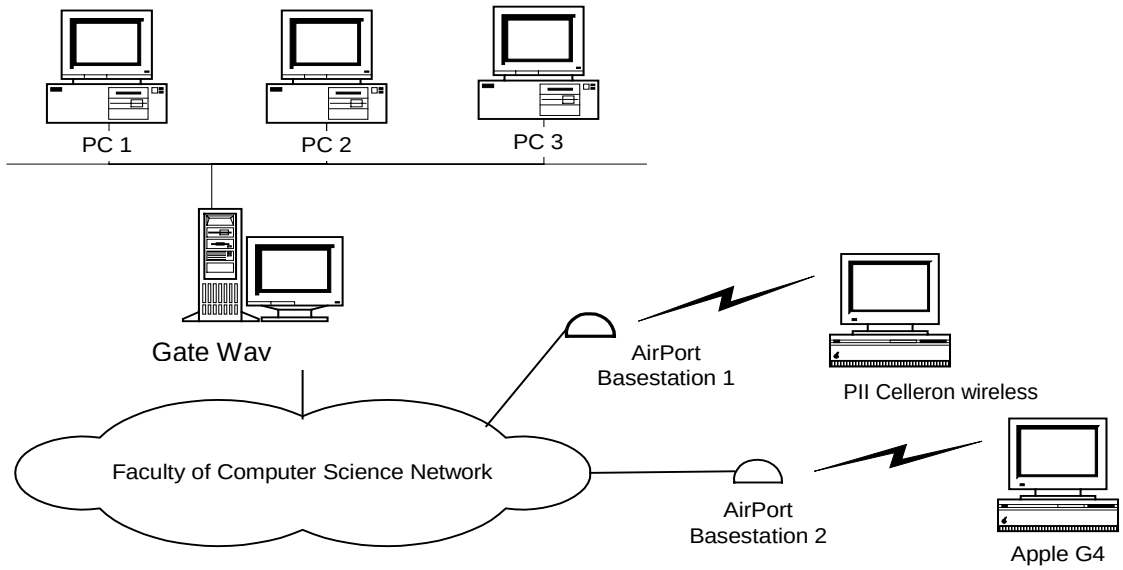


Fig.1 Network Topology

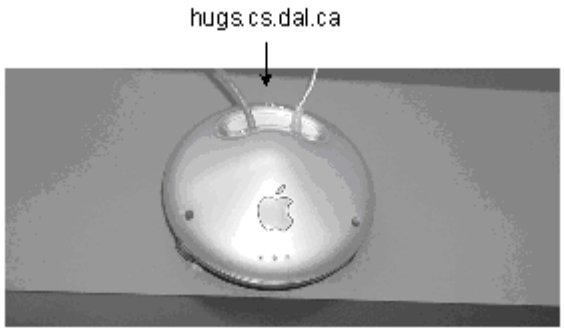


Fig. 2 hugs.cs.dal.ca AirPort (Bases Station) [14]

### 6.2 System Operation

The implemented XTR consist of two applications written in C and uses free library for very long values called lip.h, in addition to lip.h [15], segments from free implementation of XTR provided by both Lenstra and Verheul are also used [13]. The software system has been designed to work over a wireless network using the client (xtrClient.c) and server (xtrServer.c). Figure 3 and 4 illustrates both the server and the client application operations.

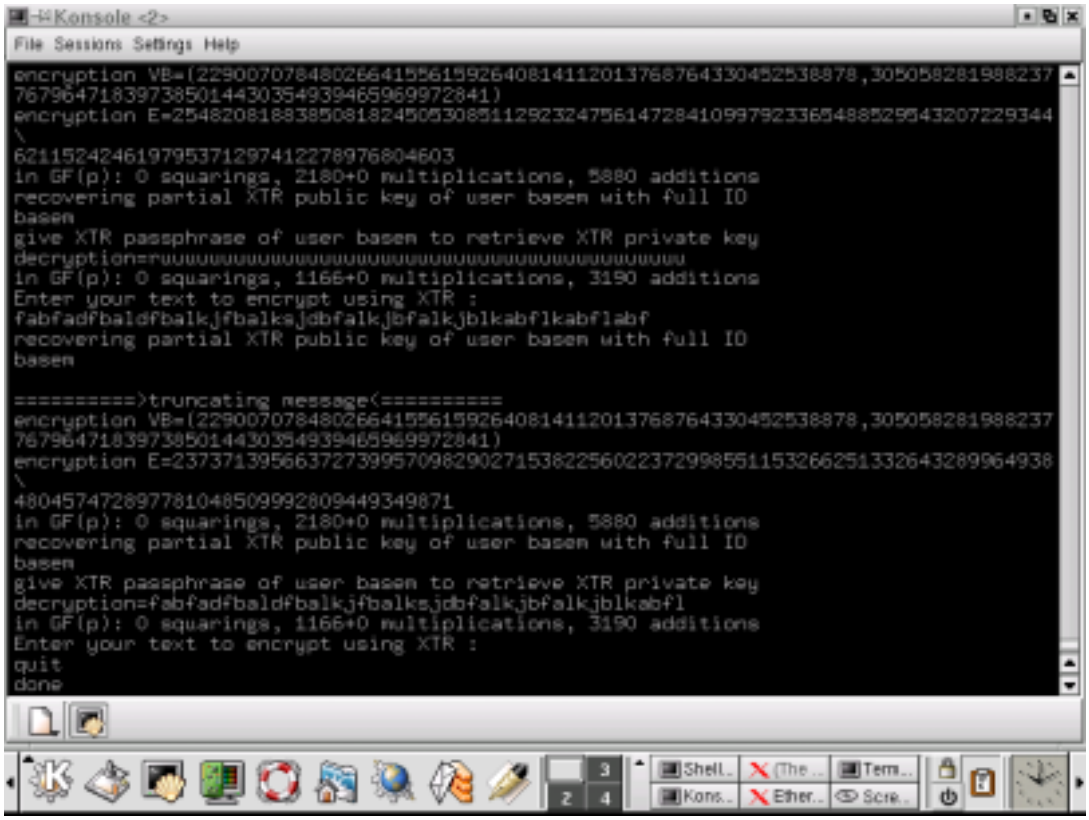


Fig.3 XTR Server Operation

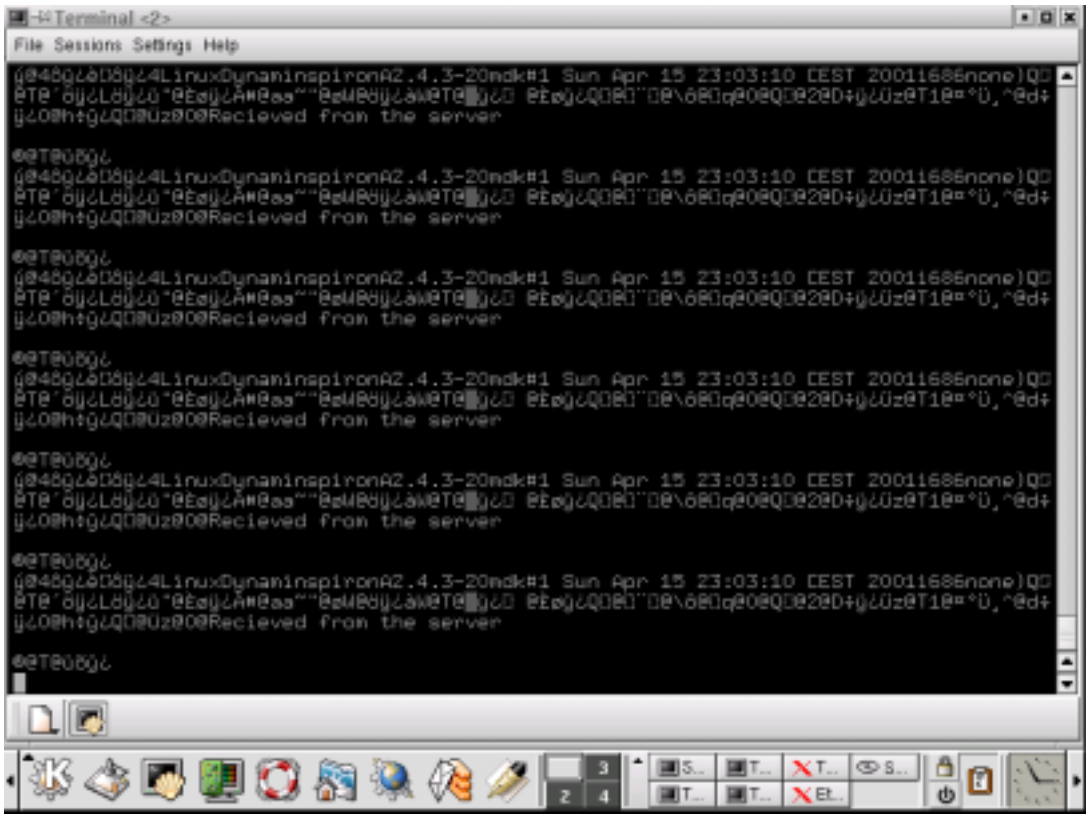


Fig. 4 XTR Client Operation

**7 Performance Analyses**

In this section the collected performance results are presented to highlight the application performance over wireless network. The performance of this design is a reflection of the overall system operations time, wireless device CPU usage and the total encryption and decryption time. Several random generated data are used to track the probability of system errors as will as bottleneck situation. Different traffic load is also generated. The following explains some experiments parameters:

1. Total number of transactions: an average of 113 different simultaneous client-server data communication has been performed.
2. Number of experiments: a total of 5 different experiments have been performed over different time periods of the day and night to get different network loads.
3. Input size: The generated text is either done through a user input or a file input. An average total of 55 different data sizes have been used to evaluate the different encryption and decryption operation time required.
4. Wireless distance: To maintain the maximum reliability of the generated results different distances have been tested (10, 50,100) meters from the AirPort (Base Station).

The following sections graphs the numerical values generated from the experiments.

### 7.1 System Enc. & Dec.

Figure 5 illustrates the encryption time. Using different sizes of input data its shown that the encryption time ranges between 30 to 60 microseconds. This test indicates how fast the XTR encryption engine over the wireless system hardware. The graph shows consistency between the data length and the used CPU encryption time.

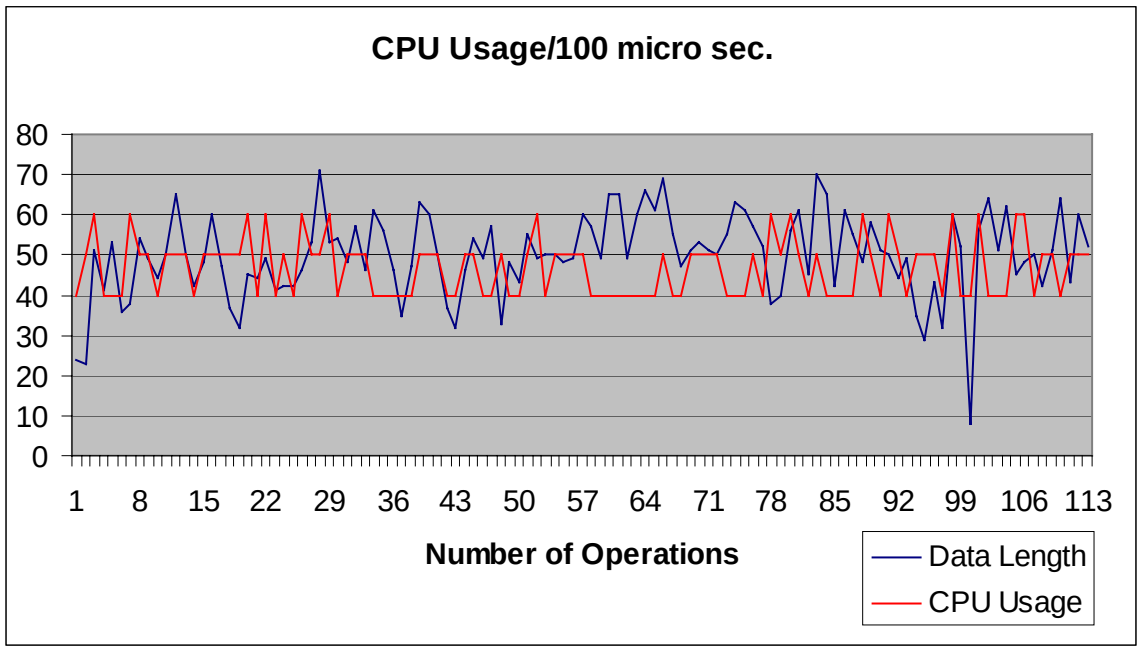


Fig. 5 CPU Usage in data Encryption process

On the other hand, using the same resulted encrypted data; figure 6 illustrates the CPU usage in executing the XTR decryption engine. It's shown that the decryption time also ranges between 30 to 60 microseconds. This test indicates how fast the XTR decryption engine over the wireless hardware system. The generated performance (in table format) results can also be found in the appendix, and the performance folder of this package.

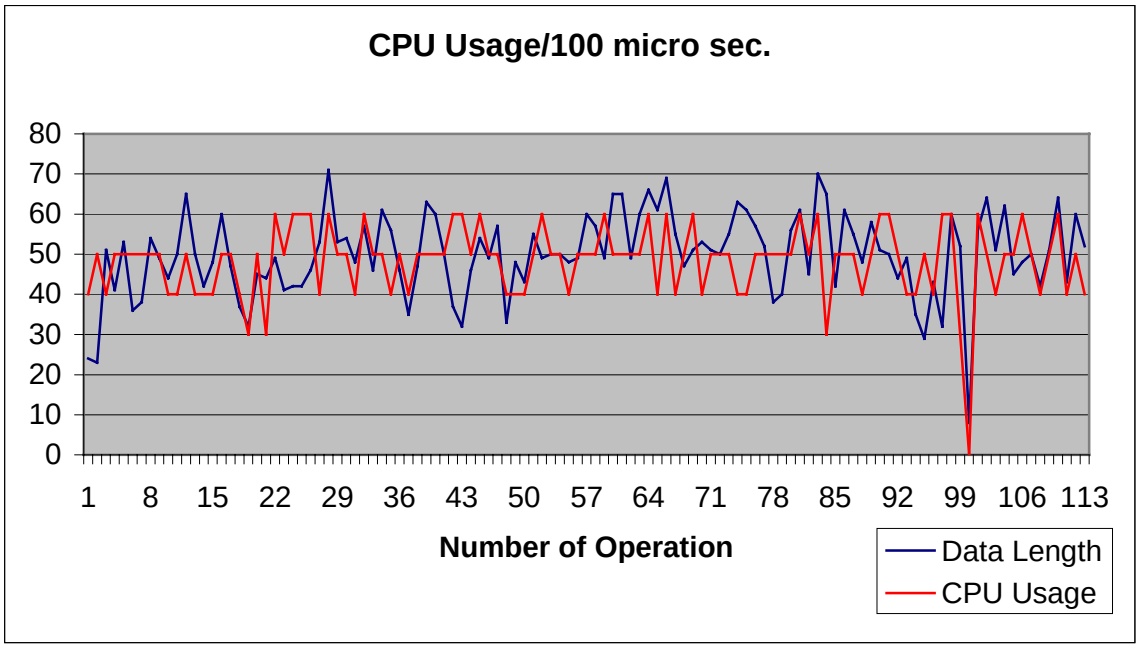


Fig. 6 CPU Usage in data Decryption process

**7.4 Security proof using Ethereal**

During the experiment, a network utility, Ethereal, is used to capture the traffic from both the server and the client. Ethereal is made to listen to port 3490 (XTR client and server port) to make sure the data does not appear as clear text, a recognizable hexadecimal pattern is sent by the network traffic generator (All F's or U's). From the snooped packets, the content of the packets in hexadecimal format is unrecognizable (no consecutive F's or U's in the contents).

Figure 7 shows Ethereal snooped data from a 10 m distance from the AirPort, while figure 8 shows the snooped data from a 100 m distance. Figure 9 shows a demonstration of ping command. Ping command has been used to make sure that there exist a packet delay between the wireless system and the AirPort base station in the 100m experiments.

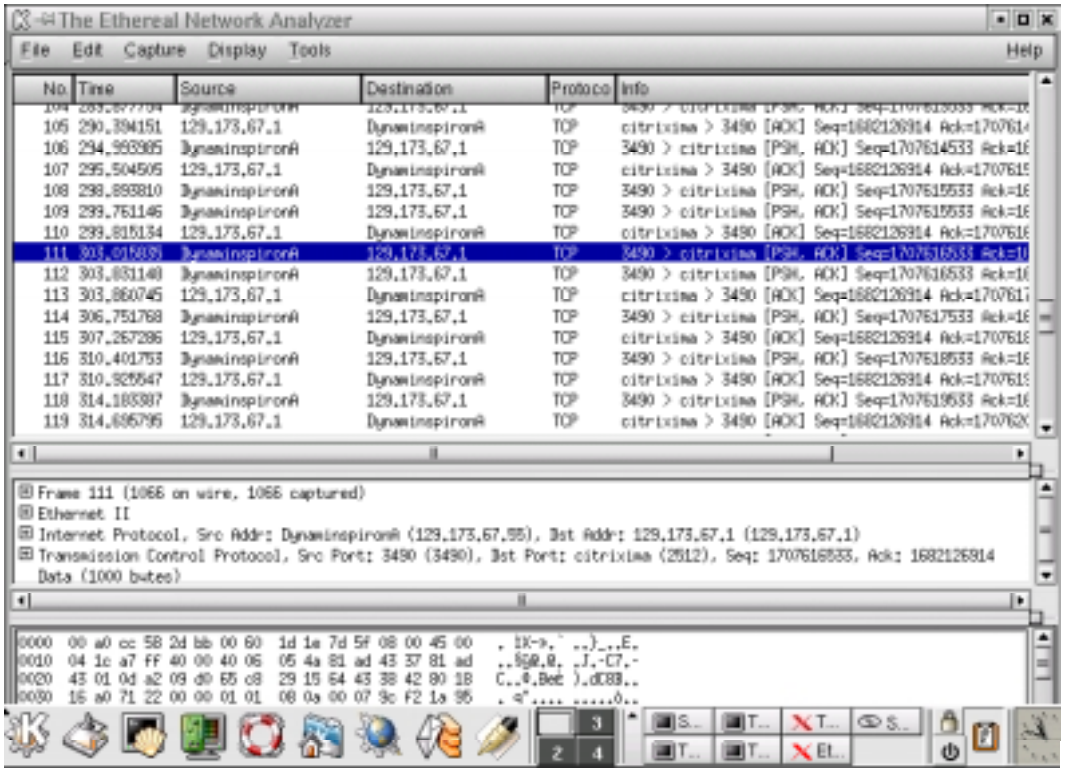


Fig. 7 Ethereal snooped data from a 10m distance

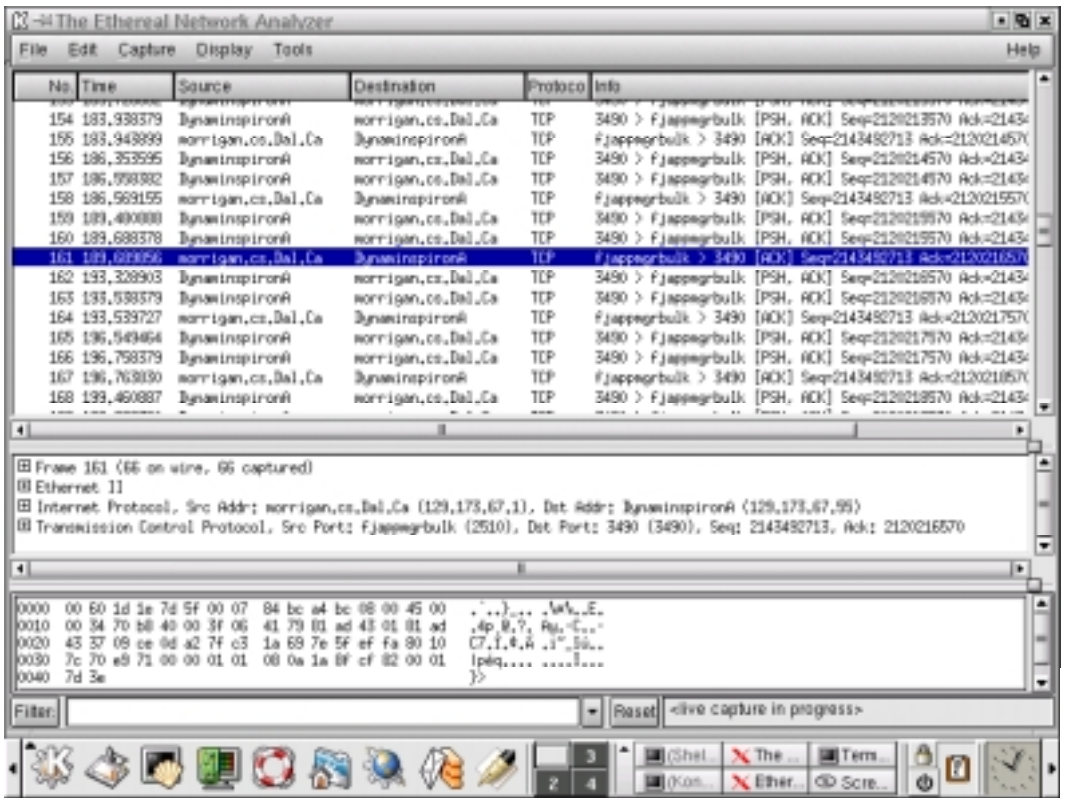


Fig. 8 Ethereal snooped data from a 100m distance

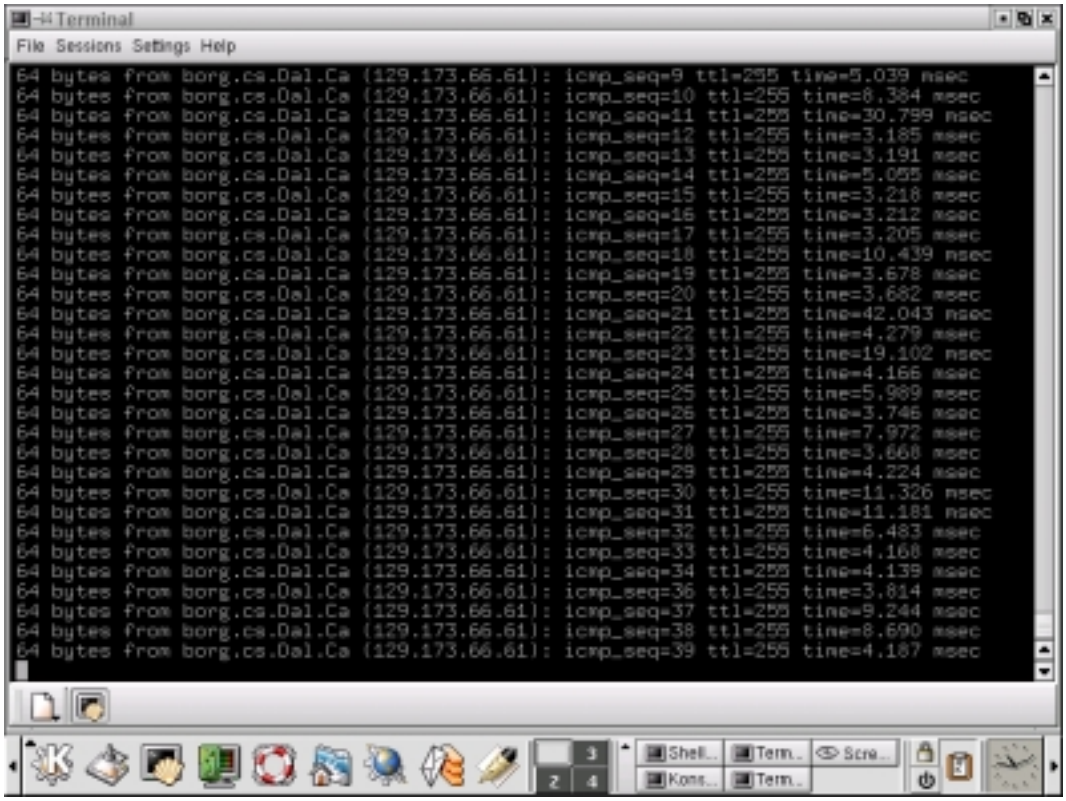


Fig. 9 Ping used for weak coverage proof

## 8 Conclusion

This project presented an implementation and performance evaluation to XTR public key cryptographic system over wireless network. The achieved goal of this project is a development of an efficient and portable secure wireless network to perform a variety of wireless applications in a secure manner.

Wireless networks technology have become an important technology in the field of computer networking. Flexibility of wireless network devices gives the network applications the flexibility needed for mobile activity. Generally, the new idea of XTR gives the wireless distributed architecture a good solution to reserving the communication bandwidth and better optimizing the computation loads, to achieve easy and light weight security deployment.

The project literately surveyed XTR mathematical, theoretical background and system implementation over wireless network. In an effort to render inherently vulnerable wireless communication more secure and faster, XTR public key cryptographic system was used in this project. XTR was used as a wireless security VPN solution to construct secure tunnel between the wireless node (server) and the weird node (client).

## 8.1 Contribution of the Project

The primary contribution of this project is an implementation and performance evaluation of XTR over wireless network. The goal of using XTR is to decrease the computation overhead and the communication loads. The following goals were achieved in the project:

1. Evaluation of XTR over wireless networks: Evaluate the performance of XTR public key cryptographic system over wireless networks from the theoretical and practical perspectives.

2. Distributed lightweight wireless security: Adopt new distributed lightweight security mechanisms that are applied over wireless network. The data, which transferred between the wireless nodes, gets encrypted with minimum operation requirements.

## 8.2 Features of XTR

The implemented XTR public key cryptosystem has provided several advantages to wireless network; the following is a description of these features:

1. Reduced overhead computations: XTR provides factor of 3 computation reduction.

2. Expanding the reach of the wireless devices: Saving communication overheads and sending light weight encrypted data saves bandwidth. This expands the wireless device reachability, allowing better bandwidth and CPU utilization.

3. Wireless device bandwidth adjustment: The factor 3 computation utilizes the wireless device computation overhead.

## 8.3 Moving from Prototype to Real World System

The prototype implemented in this project has the potential to be transformed into a complete secure product. The prototype is robust in the event of failures. For example, if a connection between the client and the server breaks down, all the pre-done operations can be retrieved because they are saved on the permanent storage each time an operation is performed. The prototype is robust in the event of security threats. The system implements data encryption and decryption techniques to ensure that the data is confidential.

However, additional work needs to be done to study the full effects of failures and bandwidth caused because of the complexity of the network environment and types of data.

### 8.4 Future Work

As an extension of this implementation, faster and shorter keys are already proposed in [16], the implementation could also benefit from these feathers. XTR cryptographic schemes for non-repudiation services, XTR signatures, and XTR DSA signatures are all very useful algorithms but not implemented in this project [8,9].

## 9 Acknowledgement

I would like to thank Dr. Giesbrecht for his insightful discussions, which lead me to fully understand ideas of symbolic computation and XTR. I would also like to thank the Secure Active VPN Environment (SAVE) group under the leadership of my master's supervisor Dr. S. Srinivas at Dalhousie University for allowing me testing this implementation on their wireless network. Also I would like to than my college J. Talor who worked with me on simulating XTR using Maple.

## 10 References


[1] U. Varshney, "*Recent Advances in Wireless Networking*" *IEEE Computer*, vol. 33, pp. 100-103, June 2000.

[2] L. Korba, "*Security System for Wireless Local Area Networks*" in *Ninth IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, 1998*, vol. 3, pp. 1550-1554.

[3] S. F. Russell, "*Wireless network security for users*", in proc. *IEEE international conference on information technology*: coding and computing, 2001, pp 172-177 April 2001

[4] Donald Davies and Wyn Price: "Security for Computer Networks", John Wiley, 1989.

[5] Bruce Schneier: "Applied Cryptography", second edition, John Wiley & Sons, 1996Cryptography book

[6] Douglas R. Stinson, "Cryptography, Theory and Practice", CRC Press, 1995

[7] SSL cryptography corner, http://www.ssh.com/tech/crypto/algorithms.cfm, last visited April 17, 2002. [Online]

[8] A. Lenstra, E. Verhuel "*An overview of the XTR public key system*", in proc. Public key cryptography and computational number theory conference, 2001

[9] A. Lenstra, E. Verhuel " *The XTR public key system*", in proc. Cryptography- Crypto 2000 lecture notes in computer science, sping-verlag. 2000 pp. 97-101

[10] E. Verhuel, "*Evidence that XTR is more secure than supersingular elliptic curve cryptosystem*", Proc. of Eurocrypt 2001.

[11] A. J. Menzes, "*Comparing the security of ECC and RSA*", manuscript, 2000, available at www.cacr.math.uwaterloo.ca/~ajmeneze [Online]

[12] A. Odlyzko, "*Discrete Logarithms*", The past and the future, Designs, Codes and Cryptography, 19 (2000), 129-145.

[13] Lenstra, Verhuel XTR crypgographic system http://ecstr.com website, last visited April 17, 2002. [Online]

[14] Apple Tech Info Library, "*Airport: Difference between DSSS and FHSS*" [Online], Available at: http://til.info.apple.com/techinfo.nsf/artnum/n58530

[15] Arjen K. Lenstra, 1989-2000, free lip.h, lip.c, long integer package, version 1.1. http://ecstr.com [Online]

[16] A. Lenstra, E. Verhuel “Key improvements to XTR”, Proc of Asiacrypt 2000, LNCS 1976, Springer-Verlag 2000, 220-233.