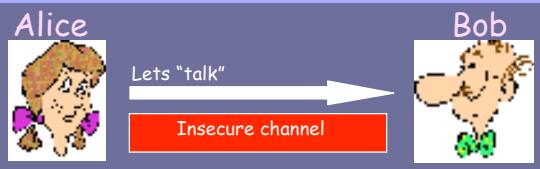
Toward Mitigating Denial of Service Attacks in Power-Constrained Sensor Networks

Ortal Arazi, Hairong Qi

College of Engineering
Dept. of Electrical & Computer Engineering
The University of Tennessee

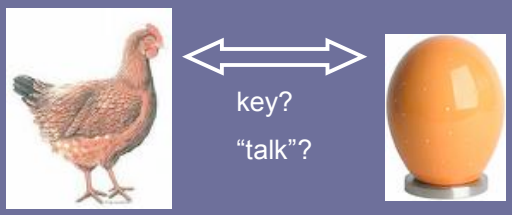
Cyberspace Sciences & Information Intelligence Research Group (CSIIR)
Oak Ridge National Laboratory, Oak Ridge

Instigating Communication



Alice and Bob: legitimate users

They would like to start communicating

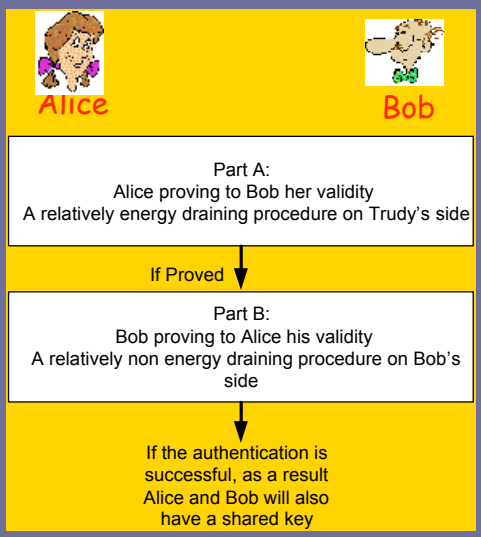


Denial of Service Mitigation

Trudy wants to:
1) Drain Bob's energy
2) Impede him from talking to other legitimate users

How can we prevent that?

Denial of Service Mitigation Procedure



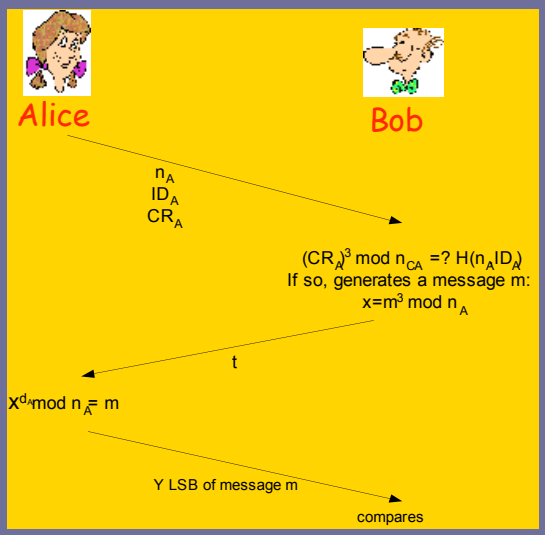
The Denial of Service Mitigation
Part A

$(n_A, e) \rightarrow$ Alice's public key
$(n_A, d_A) \rightarrow$ Alice's private key
$CR_A \rightarrow$ Alice's certificate
$ID_A \rightarrow$ Alice's identification

CR- A certificate
The CA's signature on the association between n_A and ID_A

$$CR_A = [H(n_A, ID_A)]^{d_{CA}} \bmod n_{CA}$$
$$H(n_A, ID_A) \equiv n_A \oplus ID_A$$

The Denial of Service Mitigation
Part A (cont)

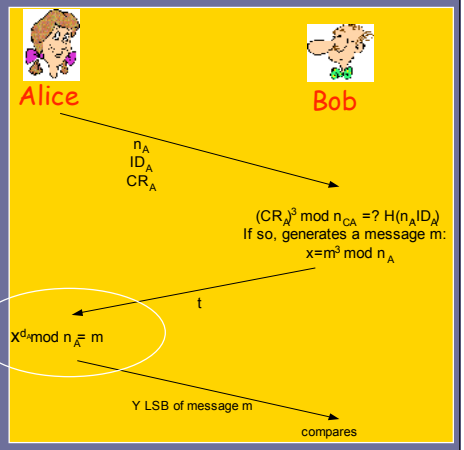


Timing and energy considerations (Part A)

$$t^{d_A} \bmod n_A = m$$

~ 160 mJ
~250 msec
(for a 512 bit key)

All other energy and time consumptions (from the other procedures) are <u>negligible</u>



Calculating the keys

(n_A,e)→ Alice's public key
(n_A,d_A)→ Alice's private key

$$n_A = p_1 * p_2$$

P_i- Pseudo random prime number

$$d_A = e^{-1} \bmod \varphi(n_A)$$

- $\varphi(n_A)$-Euler's Totient Function

Euler's Totient Function $\varphi(n_A)$ returns the number of Integers less than n_A

$$\varphi(n_A) = (p_1 - 1) \cdot (p_2 - 1)$$

Checking the Certificate

n_{CA} and d_{CA} are calculated using the exact procedure indicated above

$$n_{CA} = p_1 * p_2$$

$$d_{CA} = e^{-1} \bmod \varphi(n_{CA}) = \frac{2\varphi(n_{CA}) + 1}{3}$$

$CR_A = [H(n_A, ID_A)]^{d_{CA}} \bmod n_{CA}$

The validity procedure:

$(CR_A)^e \bmod n_{CA} \stackrel{?}{=} H(n_A, ID_A) \iff (CR_A)^3 \bmod n_{CA} \stackrel{?}{=} H(n_A, ID_A)$

$\Longrightarrow$ the calculations is the following:

$(CR_A)^3 = \left[[H(n_A, ID_A)]^{d_{CA}}\right]^3 \bmod n_{CA} = [H(n_A, ID_A)]^{d_{CA} \cdot 3} \bmod n_{CA}$.

Since $d_{CA} = 3^{-1} \bmod \varphi(n_{CA})$, $d_{CA} \cdot 3 = 1$

$\Longrightarrow (CR_A)^3 = [H(n_A, ID_A)]^1 \bmod n_{CA} = H(n_A, ID_A)$

$[H(n_A, ID_A) < n_{CA} = p_1 * p_2]$.

The Denial of Service Mitigation Procedure

Alice

Bob

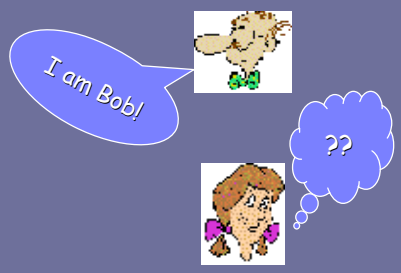
Part A:
Alice proving to Bob her validity
A relatively energy draining procedure on Trudy's side

If Proved

Part B:
Bob proving to Alice his validity
A relatively non energy draining procedure on Bob's side

If the authentication is successful, as a result Alice and Bob will also have a shared key

The Denial of Service Mitigation
Part B



How can Bob prove his validity?

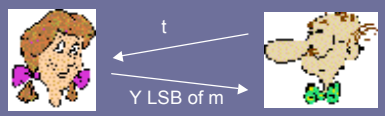
Several solutions:

1. Using the self-certified fixed key method
2. Using RSA
3. Using ECDSA

The Denial of Service Mitigation
Part B

(1) Using the self-certified fixed key method

When calculating the self-certified fixed key, Bob is authenticated!



$x_A[H(ID_B, U_B) \cdot U_B + R]$, $x_B[H(ID_A, U_A) \cdot U_A + R]$

Both have the message m Bob sent in part A

The final ephemeral key:
f(fixed key, m/without the Y LSB sent on the open channel)

Self certified DH key generation: Fixed key

Each node is given by the CA (Certifying authority) a set of public and private keys: (U_v, X_v)

Node i — ID_i , U_i

Node j — ID_j , U_j

Node i calculates: $X_i[H(ID_j , U_j)_* U_j + R]$ = $X_j[H(ID_i , U_i)_* U_i + R]$:Node j calculates

ID_v: identification of node v - scalar
U_v : node v's public key, generated by the CA - a point on the curve
X_v : node v's private key, generated by the CA - scalar

Self certified DH key generation: Fixed key

mathematical assertions …

As given by the CA:

$U_i = h_i * G$ $\qquad$ $U_j = h_j * G$

$X_i = [H(ID_i, U_i)_* h_i + d] \bmod \text{org } G$ $\qquad$ $X_j = [H(ID_j, U_j)_* h_j + d] \bmod \text{org } G$

Node i calculates:

$X_i[H(ID_j , U_j)_* U_j + R]$
$= X_i[H(ID_j , U_j)_* h_j * G + d*G]$
$= X_i[H(ID_j , U_j)_* h_j + d] *G$
$= X_i * X_j *G$

Node j calculates:

$X_j[H(ID_i, U_i)_* U_i + R]$
$= X_j[H(ID_i, U_i)_* h_i * G + d*G]$
$= X_j[H(ID_i , U_i)_* h_i + d] *G$
$= X_j * X_i *G$

R : the CA's public key = d*G - a point on the curve
d : the CA's private key - scalar
G : a generating group-point, used by all relevant nodes - a point on the curve
h_v : a random 160 bit number generated by the CA - scalar

The Denial of Service Mitigation Part B

(2) Using RSA

Message m, 512 bits		
Z	Y	x
212 bits	200 bits	100 bits

CR_B
ID_B
n_B
S_B

1. Bob calculates: [redacted]
2. Bob send Alice: [redacted]
3. Alice calculates: [redacted]

If so, the final ephemeral key: **Z**

The Denial of Service Mitigation Part B

(3) Using ECDSA

Bob:

- Generates a random number: u. Calculate [redacted] .
  C- the scalar representation of point V.
- Calculates [redacted] .
  The signature is the pair (C,L)
- Sends Alice (C,L)

Message m, 512 bits		
Z	Y	x
212 bits	200 bits	100 bits

If so, the final ephemeral key: **Z**

Alice:

- Computes: [redacted]
- Obtains the curve point: [redacted]
  C'- the scalar representation of point P

If C=C', then the signature is valid, it is Bob!

The Denial of Service Mitigation, Part B
Comparing the three methods

The time is measured in: ECC point by scalar multiplications
Approximately: 40 msec

	Method 1 (Fixed key)	Method 2 (RSA)	Method 3 (ECDSA)
Part A Alice proving her identity	Alice: 6 Bob: ~0	Alice: 6 Bob: ~0	Alice: 6 Bob: ~0
Part B Bob proving his identity	Alice: 2 Bob: 2	Alice: ~0 Bob: 6	Alice: 2 Bob: 1
Overall computational overhead	~10	~12	~9

Conclusions

- PKC implementation in WSN is feasible
- ECC shows promise as crypto technology
- DoS is a primary threat
- Introduced a hybrid RSA/ECC framework for mitigating DoS attacks
- Using the fixed key approach or the ECDSA approach proved to be highly beneficial

Thank You

Questions ?