Design Space Exploration for Energy-Efficient Secure Sensor Network

Lin Yuan and Gang Qu
Electrical and Computer Engineering Dept. and Institute of Advanced Computer Study
University of Maryland, College Park, MD 20742
{*gangqu,yuanl*}*@glue.umd.edu*

Abstract

We consider two of the most important design issues for distributed sensor networks in the battlefield: security for communication in such hostile terrain; and energy efficiency because of battery's limited capacity and the impracticality of recharging. Communication security is normally provided by encryption, i.e., data are encrypted before transmission and will be decrypted first on reception. We exploit the secure sensor network design space for energy efficiency by investigating different microprocessors coupled with various public key algorithms. We propose a power control mechanism for sensors to operate at an energy-efficient fashion using the newly developed dynamical voltage scaling (DVS) technique. In particular, we consider multiple voltage processors and insert additional information into the communication channel to guide the selection of proper voltages for data decryption/encryption and processing in order to reduce the total computational energy consumption. We experiment several encryption standards on a broad range of embedded processors and simulate the behavior of the sensor network to show that the sensor's lifetime can be extended substantially.

1 Introduction

Distributed Sensor Network (DSN) will produce high-quality information for both civil and military applications using large number of physical sensors (e.g., acoustic, seismic, visual) communicating via ad hoc wireless networking. Advances in digital circuitry, wireless communications, battery technology, and microelectromechanical systems (MEMS) technology make sensors smaller, less expensive, more versatile, reliable, and durable. Energy efficiency is among the most interesting challenges in DSN design[6, 11]. This is because of the difficulty of power-source maintenance for the sensors, in particular in a hostile environment such as battlefield. In many occasions, the durability of the DSN is defined as battery's lifetime. Meanwhile, providing confidentiality and authentication is also critical for such applications to prevent an adversary from compromising the security of the DSN.

In this paper, we address these two secure DSN design issues at system level. More specific, we take advantage of the multiple voltage design methodology to reduce the (computational) energy consumption in the sensor network based on the following two observations:

- **Unbalanced computation load for decryption and encryption:** the computation load (and hence execution time and energy consumption) for encryption and decryption are quite different in most asymmetric public key cryptographic algorithms such as RSA and El Gammal. For example, 145,408 128-bit multiplications are required by RSA decryption, comparing to only 7,056 by the RSA encryption.
- **Large variety of data processing requirement:** on the reception of an (encrypted) message, the sensor node has to first decrypt the message. However, the time and energy to process (excluding encryption) the message can be and they may not be proportional to the length of the message. For instance, multi-hop communication is widely used because of the high energy cost of long-distance transmission[1, 6, 19, 20]. If sensor A asks sensor B to forward a message to another sensor node, then B does not need to do any data processing.

Proceedings of the IEEE International Conference on Application-Specific Systems, Architectures, and Processors (ASAP'02)
1063-6862/02 $17.00 © 2002 **IEEE**

We propose to apply the newly developed dynamical voltage scaling (DVS) technique in the design of energy-efficient DSN. DVS is a technique that varies the supply voltage and clock frequency based on the computation load to provide desired performance with the minimal amount of energy consumption. We consider a practical DVS system which is capable of switching among several simultaneously available voltage levels. To fully take advantage of the DVS system, we insert additional information about the message called *message header* at the beginning of each message from the sender sensor. The message header contains information such as length of the message, type of the message, expected processing time, length of the result, and deadline among others. The sensor node on the receiving end can utilize this information to properly select voltages for data decryption/encryption and processing to reduce the overall energy consumption.

2 Related work

Distributed sensor networks (DSN) have attracted a lot of attentions in recent years. Researches in many areas (MEMS, wireless communication, network, cryptography, just to name a few) have addressed the design and implementation issues of DSN from their point of views.

The Wireless Integrated Network Sensors (WINS) project at UCLA[18] and Rockwell Science Center[19] develops low power, low cost, wireless MEMS-based microsensors that can sense, actuate, and communicate. Power efficiency is provided by power management over the network, low power mixed signal circuits, and low power radio frequency (RF) receivers[2]. Berkeley's "Smart Dust" project[20] uses optical (instead of RF) transmission techniques to make communication inexpensive energy-wise[11]. The Ultra Low Power Wireless Sensor Project in MIT targets the design and fabrication of sensor systems that is capable of wirelessly transmitting data at 1 bit to 1 megabit per second with average transmission power 10 microwatts - 10 milliwatts[21]. They focus on developing energy-efficient communication protocols, in particular, energy-scalable algorithms[7, 16].

Dynamically adapting voltage and therefore the clock frequency, to operate at the point of lowest power consumption for given temperature and process parameters was first proposed in early 90's [10, 12]. The implementation of several digital power supply controllers [8] and efficient DC-DC converters that allow the output voltage to be rapidly changed under external control[13] were reported shortly. Microprocessor systems that are capable of changing its speed dynamically were implemented recently. The lpARM processor [3, 14] is based on the ARM8 core and designed to operate between 1.1v and 3.3v, resulting in speeds between 10MHZ and 100MHZ. Clock frequency transition take approximately $25\mu s$ (about 1250 cycles) for a complete 10MHZ to 100MHZ transition.

3 DSN Design space exploration

A distributed sensor network (DSN) consists of a collection of communicating nodes, where each node incorporates a) one or more sensors for measuring the environment, b) processing capability in order to process sensor data into "high value" information and to accomplish local control, and c) a radio to communicate information to/from neighboring nodes and eventually to external users[19].

3.1 Model of sensor node and energy consumption in a secure DSN

We consider a DSN in a hostile environment where the communication among sensor nodes must be secure. Security is provided by standard data encryption protocols with pre-established keys.

In a secure DSN, sensor nodes obtain information locally from target detection or environment monitoring. However, the major information source is the communication channel in the network. Encrypted data packet from neighbor sensor nodes is received by the reception electronics and passed to the microprocessor. Data decryption and verification (to ensure that the data comes from an authenticated node) are conducted first before data processing is performed. If the result needs to be sent out to other sensor nodes, the microprocessor will first encrypt the result, then send it to the transmission devices and halt. Otherwise, when this

Proceedings of the IEEE International Conference on Application-Specific Systems, Architectures, and Processors (ASAP'02)
1063-6862/02 $17.00 © 2002 IEEE

sensor node is the final recipient of the data and do not need to cooperate with other nodes, the microprocessor will halt immediately after data processing. The transmission devices, normally consists of transmission electronics and amplifiers, will send the data packet out.

The energy dissipation through the sensor network is the sum of that consumed by all the sensor nodes in the network. This includes three parts: (1) energy dissipation on the sensor transducer; (2) energy dissipation for the communication among sensor nodes; and (3) energy consumed by the microprocessor on computation. The amount of energy consumed by the sensor transducers depends on the sensibility of the sensor and is normally a minor part of the entire sensor node's energy consumption[1]. Radio transmission may contribute more than half of the peak power. It consumes more power at transmit mode than receive mode because the transmit amplifier must be active at transmitter's end. Radio transceivers are relatively complex circuits and it is difficult to reduce the communication energy consumption. A simplified model for communication energy can be found in [7] and techniques to minimize this part of the energy consumption can be found in [2, 11, 18, 21] and is beyond our scope.

Requirements and constraints need to be addressed during the design of DSN include: embedded processor selection and memory design in architecture layer; routing, scalability, and robustness of at network layer; real-time operating systems and power-aware software at application layer; transmission technologies, sensor fabrication and deployment issues. In addition, a secure DSN has to support the following features for security concerns: low energy consumption, confidentiality, authenticity, unidirectional communication, and tamper resistance.

Our target is to select embedded microprocessor and data encryption protocols such that the energy consumption on the microprocessor will be minimized.

3.2 Energy consumption for data encryption protocols on multiple voltage processors

Data encryption and decryption play the vital role in secure DSN. Therefore it is useful to measure the energy consumption for different data encryption/decryption algorithm on different embedded microprocessors. Table 1 gives the computational energy cost for RSA encryption/decryption, DSA data signing/verification, and ElGamal encryption/decryption on five microprocessors. It is obtained by first estimating the energy cost for computing the 128-bit multiply function, the basic building block for most data encryption protocols, at the reference $3.3v$, and then multiplying by the number of 128-bit multiplications required by each computation (see [4] for details).

processor	Power	Frequency	Energy Consumption (mJ)					
	(mW)	(MHz)	RSA		DSA		ElGamal	
MIPS R4000	230	80	16.7	0.81	9.9	20.0	9.94	134
SA-1110 "StrongARM"	240	133	15.0	0.74	9.1	18.2	9.1	123
MC68328 "DragonBall"	52	16	840	42	520	1040	520	7000
MMC2001 "M-Core"	81	33	137	6.9	85	169	85	1140
ARC 3	2	40	1.13	0.06	0.70	1.40	0.70	9.4

Table 1. Energy consumption for data encryption protocols at $3.3v$ (the two values in "Energy Consumption" column are decryption/encryption for RSA and ElGamal, and data verification/signing for DSA).

Processor's dynamic power dissipation is proportional to capacitance, clock frequency, and the square of supply voltage ($P \propto C_L \cdot v_{dd}^2 \cdot f$). Suppose that P_i is the power consumption at voltage v_i ($P_i \propto v_i^2$), f is the clock frequency ($f \propto v_i$), and t_i is the CPU time to complete the same workload as the processor running at certain reference voltage in 1 CPU unit ($t_i \propto 1/v_i$), the energy consumption is given by $E_i = P_i \cdot t_i$, which

[1]For example, in the "AWAIRS I" sensor developed at UCLA/Rockwell Science Center, the entire sensor node consumes a peak of 1W of power. The processor consumes 300mW, the radio consumes 600mW at transmit mode and 300mW in receive mode. The sensor transducers consumes less than 100mW[1].

Proceedings of the IEEE International Conference on Application-Specific Systems, Architectures, and Processors (ASAP'02)
1063-6862/02 $17.00 © 2002 IEEE

is proportional to v_i^2 roughly. This implies that to accumulate the same amount of computation, using lower voltage will consume less energy in longer time because the power level is much lower.

v_{dd} (volt)		3.3		2.4		1.2	
power (mW)		230		82		7.5	
clock frequency (MHz)		80		54		20	
		t(ms)	energy (mJ)	t(ms)	energy (mJ)	t(ms)	energy (mJ)
RSA	decryption	72.7	16.7	107.6	8.8	290.4	2.2
	encryption	3.5	0.81	5.2	0.43	14.1	0.11
DSA	data signing	43.1	9.9	63.8	5.2	172.2	1.3
	verification	87.0	20.0	128.8	10.6	347.8	2.6
ElGamal	decryption	43.2	9.94	64.0	5.25	172.9	1.3
	encryption	582.7	134	863.1	70.8	2330	17.5

Table 2. Performance and energy consumption under multiple voltages.

Assuming each processor is implemented with three different supply voltages ($3.3v, 2.4v,$ and $1.2v$), one can calculate the power consumption and clock frequency at different voltages. We report the performance and energy consumption for public key encryption/decryption on MIPS R4000 in Table 2. Our power control mechanism selects the lowest possible voltage to reduce energy consumption.

3.3 The message header

To enable power control for energy efficiency, sensor node at the receiving end must have additional information about the upcoming message to avoid selecting inappropriate voltages[2]. The earlier such information is available, the better decision can be made, and the more energy can be saved by switching to the proper supply voltage. The complete knowledge of the message will not be available to the receiver sensor until it is completely decrypted. However, this may be already late for energy reduction because data decryption consumes energy and for some public key cryptographic algorithms such as RSA this energy consumption is significant.

On the other hand, the sender sensor knows the message better than the receiver before the messages is completely decrypted. Therefore, we propose to have the sender sensor collect additional information about the message and send it out as the *message header*.

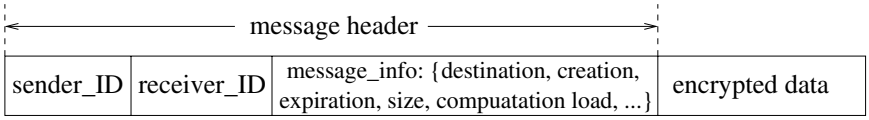

Figure 1. Content of the first data packet in a message.

As depicted in Figure 1, the message header is embedded in an encrypted and signed (for authentication purpose) data packet, which is the first of a message and contains the followings:

- sender's information such as the sensor's ID that the receive sensor can verify to avoid attacks.
- receiver's information so that a receive sensor can tell whether he is the desired receiver of the message (for multicast).
- message's information, which is the key part of the message header. It includes: size of the message either in bits or in number of packets, time the message is generated and its latency requirement both specified in a global clock, estimated computation load for data processing in CPU time at a reference

[2]If the selected voltage is higher than necessary, then further energy reduction is still possible; if the selected voltage is lower, then the sensor is in danger of missing the deadline or has to raise voltage level later on to catch it, which will increase the overall energy consumption.

Proceedings of the IEEE International Conference on Application-Specific Systems, Architectures, and Processors (ASAP'02)
1063-6862/02 $17.00 © 2002 **IEEE**

voltage, predicted destiny of the message (e.g., being forwarded to other sensor nodes or not, being sent feedback to senders or not), etc.

- (part of the) data to be sent if there is still space left in the data packet after holding all the above information.

Since most of these information require little space[3], including the message header will introduce one extra packet in the worst case. The energy consumed on computing and transmitting this extra packet will negate some portion of the energy saving by our proposed technique. However, we will show this is insignificant.

3.4 Power management on sensor node

Figure 2 gives the overview of how microprocessor controls power by switching supply voltages.

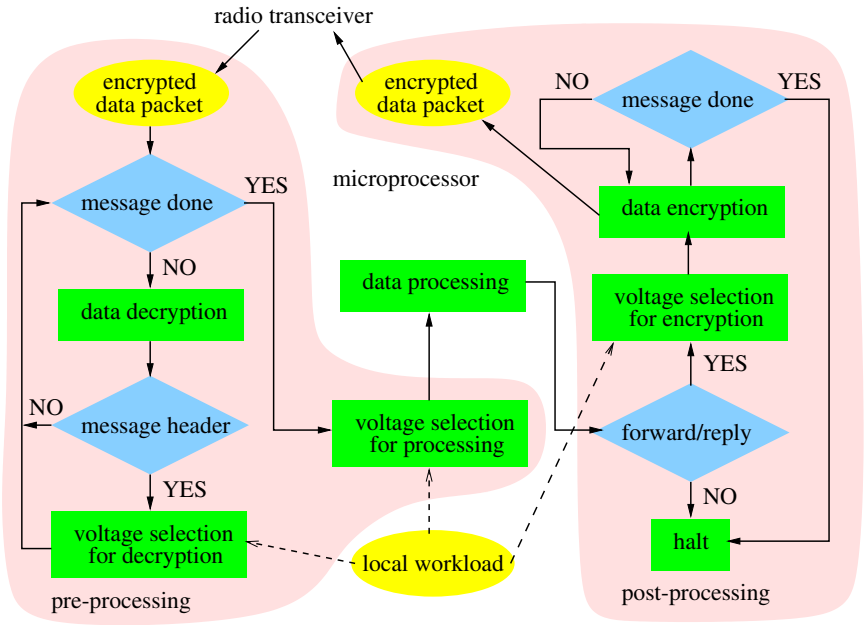

Figure 2. Power control mode in a sensor node.

The encrypted data packets received by the radio transceivers are passed to the microprocessor which decrypts and authenticates the data at the current voltage[4]. The microprocessor then check whether the packet contains a message header, if not, it continues message decryption for the following packets; otherwise, the microprocessor obtains information about the size of the message, the estimated processing load, and size of the result from the message header. Let W_{de}, W_{en}, W_{pr}, and W_{lo} be the computation workload for decrypting one packet, encrypting one packet, processing the current message (estimation by the sender sensor), and the jobs currently running on the receiver sensor node. Suppose the message has k packets and an estimated encrypted result of $\bar{k}$ packets to be completed by time t_d, the total workload needs to be finished by t_d will be $k \cdot W_{de} + W_{pr} + \delta \cdot \bar{k} \cdot W_{en} + W_{lo}$, where $\delta = 0$ if there is no need to forward the result and otherwise $\delta = 1$.

The goal of pre-processing is to decrypt the message using the most energy-efficient voltage and determine the voltage for data processing. The decrypting voltage is decided based on the information provided by in the message header and the microprocessor will decrypts the remaining packets of the message at this voltage. Once data decryption is done, the processor gains complete knowledge of the data and can update

[3]For example, sender can use two bits to encode whether the message need to be forwarded: *00 do not forward, 11 forward, 10/01 not sure, depends on the processing result.*

[4]If the microprocessor is at sleep mode, it will be woke up by an interrupt and set voltage at the default level.

Proceedings of the IEEE International Conference on Application-Specific Systems, Architectures, and Processors (ASAP'02)
1063-6862/02 $17.00 © 2002 **IEEE**

the voltage for data processing in a similar fashion. This completes the pre-processing and microprocessor will start processing the data with the selected voltage.

After data processing, the microprocessor will halt if there is no need to forward the result to other sensor nodes. Otherwise it will construct the message header, re-evaluate the size the encrypted result $\bar{k}$, and select a proper voltage for data encryption. The encrypted data goes to the radio transceiver and will be sent out. We refer this as post-processing.

4 Simulation

4.1 Simulation platform

We simulate the behavior of a sensor node in a DSN assuming that the interarrival time of messages follows exponential distribution with parameter μ; with probability α, the sensor only needs to forward the message without any data processing; the (predicted) non-zero processing requirement is uniform between e_1 and e_2; the length of the message is uniform between a_1 and a_2; the length of the (predicted) result is uniform between b_1 and b_2.Each message is non-preemptive and has a deadline. In our simulation, we use 1.5 times the time from the arrival of one message untill the start the next as message deadline; this is equivalent to requiring a maximum of one message buffer. When a message misses its deadline, it has to be dropped.

We generate a sequence of messages using the above parameters and conduct two simulations for each type of processor reported in Table 1. First on the traditional processor operating at $3.3v$ and then on the same processor with three voltages $3.3v, 2.4v$, and $1.2v$. The traditional processor will decrypt the message, process the data, encrypt the result and send it out if necessary. It remains idle and does not consume any energy when there is no message to process.

For the same sequence of messages, we assume that the sender sensor has constructed a 256-bit message header for each message and built the first packet by encrypting the 256-bit message header and the first 768 bits of the message. The rest of the message is encrypted to 1024-bit packets as before. Clearly, we may experience an overhead no worse than one packet per message. The multiple voltage processor will process the messages with headers as we described in Figure 2. Remember that a 256-bit message header will be added to the result when forward or reply is required. This may also introduce a one-packet overhead.

After decrypting the message header, the receiver sensor may take one of the following actions:

- Rejection: the receiver is not the designated receiving node(s) or the message is obsolete. The receiver will simply drop the rest of the message without further decryption[5].
- Forwarding: the receiver is asked to forward the message, which happens in multi-hop network. The receiver decrypts the entire message and encrypts it for forwarding. The decryption is based on the key agreement with the sender sensor, and the encryption is based on the key agreement with the sensor that will receive the forwarded message. The processing time in this case is 0 as we mentioned earlier.
- Acceptance: the receiver decrypts the message, does the required process, and makes decisions without sending any messages out to other nodes, including the sender of the current message.
- Proceeding: the combination of Acceptance and Forwarding. The processing result needs to be encrypted and sent back to the sender or forwarded to other nodes.

We monitor the system's energy consumption on message decryption/verification, data processing, and data encryption/signing for both simulations. In the multiple voltage case, we also keep track the total time that the system is operating at different voltages. Finally, we repeat this for each different combination of microprocessor and public key algorithms.

[5]For a fair comparison, we assume that on the traditional fixed voltage processor, the rejection can also be detected after the decryption of the first packet. This is true because the non-designated receivers will not be able to decrypt the packet and information such as expiration time is widely used in most DSNs.

Proceedings of the IEEE International Conference on Application-Specific Systems, Architectures, and Processors (ASAP'02)
1063-6862/02 $17.00 © 2002 IEEE

4.2 Simulation results

We first describe the messages that the sensor receives in the simulated communications. Then for different microprocessor coupled with different public key algorithm, we report the average energy consumed by the traditional fixed voltage processor and the multiple voltage processor for the same sets of messages. To better analyze where the energy saving comes from, we give the detail time and energy data for the case of MIPS R4000 processor with RSA. We also conduct several other case studies for questions such as what is the role of different public key algorithms and eventually how to guide the system configurations.

We simulate the behavior of one sensor node in the DSN that receives messages with the following parameters: the interarrival rate of messages $\mu = 0.125$; forwarding-only probability $\alpha = 0.5$; the range of non-zero processing time is $[500ms, 4000ms]$; both original message and processing result are of size [200 bits, 20000 bits] (i.e., 1 to 20 packets). As we have discussed earlier, a message may be rejected by the receiver, we set the rejection rate to be 0.2. On average, the sensor node receives 464 messages in one hour. Among them, 97 (21.0%) are rejected, 229 (49.4%) are forwarded without processing, the rest 137 (29.6%) require data processing. The average overhead on the received message is 118 packets, which is about 25.5% of the total messages as expected[6]. The average overhead on the result is 92 packets, 24.9% of the messages that need to be forwarded or replied. Notice that each message carries at most one extra packet overhead because of the message header.

4.2.1 Simulation on different system configurations

For one set of the messages generated from a one-hour simulation described as above, we further simulate on different combinations of microprocessors and public key algorithms. For each system, we conduct simulations on the fixed voltage ($3.3v$) core and the core with multiple voltages ($3.3v, 2.4v$, and $1.2v$). Figures 3 and 4 report the energy consumption and non-idle time for five representative systems, namely *MIPS R4000 with RSA, MIPS R4000 with DSA, StrongARM with RSA, StrongARM with DSA*, and *M-Core with RSA*. (The exclusion of the ElGamal algorithm is due to its high encryption cost.).

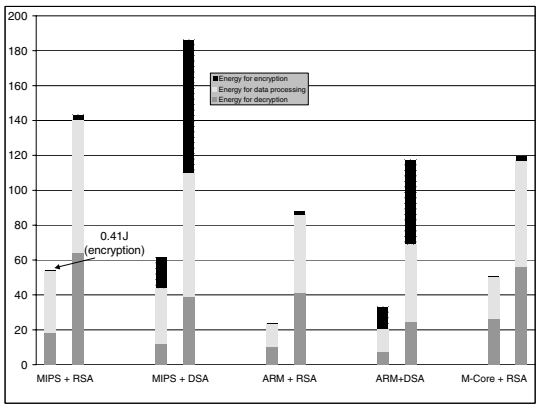

Figure 3. The energy consumption break-down on six different systems.

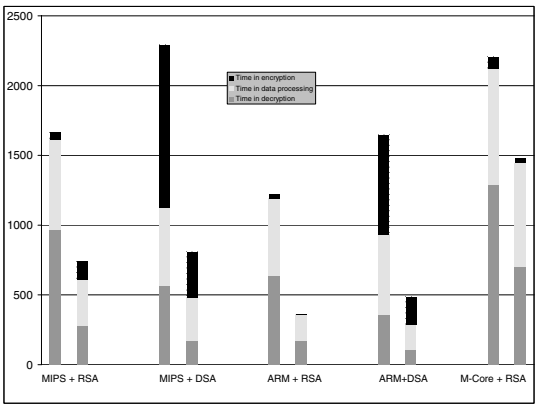

Figure 4. The non-idle time break-down on six different systems.

Figure 3 indicates significant energy reduction on all systems, from 58% in the M-Core system to 73% in the StrongARM core, with an average of 64% energy saving. Considering that the microprocessor is responsible for 30% of the sensor node's total energy consumption, this means a energy saving of 20% which is still significant. Although we see a constant 55% energy reduction from data processing, energy savings from data decryption and data encryption are very different. For RSA, where decryption is 20 times more expensive than encryption, the amount of energy saving from decryption is much more significant than

[6]The size of the message header is 256 bits and a packet is of a fixed 1024-bit. The one-packet overhead occurs if the last packet contains more than 576 bits data and hence not sufficient space left for the message header. Statistically, this happens 25% of the time.

Proceedings of the IEEE International Conference on Application-Specific Systems, Architectures, and Processors (ASAP'02)
1063-6862/02 $17.00 © 2002 **IEEE**

that from encryption. We observe the same result for the verification-expensive DSA. However, it is not so significant as in RSA because the verification in DSA is only twice as expensive as data signing. For systems with the same public key algorithm but different microprocessor, the energy saving depends on the system's power/frequence performance. For slow core such as M-Core, its energy saving for all three phases of the message processing is not so dramatic as that for fast core StrongARM. The reason is that we cannot reduce the speed on slow cores due to the message's density and their data processing requirement.

Similar analysis holds for the system's non-idle time as illustrated in Table 4. We see the multiple voltage microprocessors have longer running time than the traditional fixed cores. This suggests that to complete the same workload, it is better to run longer time at lower voltage, which is a well-known fact in dynamical voltage scaling literatures(see for example, [9, 17]).

4.2.2 Where does the energy saving come from?

	time		energy	
	fixed (s)	DVS (%)	fixed (J)	DVS (%)
decryption	276.11	350%	63.51	28.8%
encryption	13.11	413%	3.02	13.7%
processing	343.19	188%	78.93	45.0%
total	632.41	264%	145.45	37.3%

Table 3. The average time and energy consumption for the traditional fixed voltage and new multiple voltage processor in 10 simulation.

	3.3 V		2.4 V		1.2 V		total	
	time (s)	energy(J)	time (s)	energy(J)	time (s)	energy(J)	time (s)	energy(J)
decryption	34.30	7.89	44.91	3.68	888.26	6.69	967.47	18.26
encryption	0	0	0.084	0.007	54.02	0.41	54.11	0.413
processing	8.45	1.94	386.83	31.71	250.07	1.88	645.35	35.53
total	42.75	9.83	431.82	35.39	1192.35	8.97	1666.93	54.2

Table 4. Run-time and energy consumption break-down for the simulation on MIPS R4000 with multiple voltages.

We report the 10 simulations on the MIPS R4000 using RSA as the public key algorithm in detail for the analysis of the energy saving. In the traditional approach, the processor is on for $632s$ ($276s$ for decryption, $343s$ for data processing, and $13s$ for encryption) at the fixed $3.3v$ and hence consumes $145J$ energy (see Table 3 for details.). Notice that data decryption and processing take about 44% and 54% of the total non-idle time and energy. This is because the following reasons: 1) the RSA encryption is much less computation intensive comparing to decryption (Table 2); 2) result encryption is necessary only for messages that require to be forwarded or replied; 3) for rejected message, the first packet needs to be decrypted before the rejection.

In our new approach with the multiple voltage processor and message header, the insertion of additional information introduces one extra packet for 92 messages. The multiple-voltage processor decrypts the message header at the default $3.3v$ and then selects the proper voltage for the rest of the decryption. It will also drop all the 97 rejected messages at this phase without any further decryption. Once the decryption is completed, the processor update the voltage for data processing and/or result encryption. The total non-idle time is $1667s$ with an energy consumption of $54.2J$, a 62.7% energy saving over the traditional processor.(See Table 3 and 4)

Clearly we see that the multiple voltage processor spends most of time at the low 1.2v (about 72%) and 2.4v (more than 25%) to save energy. It operates at the high 3.3v only when necessary for the decryption

Proceedings of the IEEE International Conference on Application-Specific Systems, Architectures, and Processors (ASAP'02)
1063-6862/02 $17.00 © 2002 IEEE

of all first packet and the decryption and processing for messages that have a high computation load which cannot be accomplished at a lower voltage. It never does encryption at full speed. This is a coincidence, however it is very rare to encrypt at the highest voltage because the RSA encryption scheme is so cheap as we have witnessed from Table 3 and 4, that it only happens when the data decryption and processing load is high and there is not sufficient time left for encryption to be performed at a lower voltage.

4.2.3 Energy-driven system configuration

For a given set of messages with certain statistical information, it will be interesting to see what can guide us to select the right combination of microprocessor and public key algorithm to implement the secure DSN in the most energy-efficient way. For this purpose, we conduct the following simulations where for each different settings of messages, we simulate the energy consumption on all possible system configurations. The inter-arrival rate μ takes value from the set of $\{0.125$,0.1, 0.05,0.025, $0.01\}$; the message's size ranges from one of the followings: {[200,20000], [200, 4000], [200, 10000], [10000,20000]}; the processing time falls into one of the followings: {[500,4000], [100,1000],[2000,10000]}; and finally we use one of the following three sets for rejection rate, forward-only rate, and processing-demand rate: {(0.2,0.5,0.3), (0.1,0.3,0.6),(0.1,0.6,0.3)}.

We have several interesting observations from the preliminary results. For example, if the microprocessor is not fast enough to keep in pace with the message arrival rate, then it has to drop some messages due to the deadline requirement. We see in the traditional system settings, where there is no multiple voltages and no message header, for the highest inter-arrival rate $\mu = 0.125$ with moderate message size and data processing requirement, only a few system configurations consistently finish all the processing without any message drop(e.g., MIPS/SA-1110). As the message arrival rate decreases, more and more system configurations can handle the messages, while the combination of SA-1110 and RSA remains the most energy efficient. When the inter-arrival rate reaches 0.05, the most power efficient microprocessor ARC3 is able to handle the messages and becomes the best choice for DSN implementation.

Similar behavior can be found for rejection rate, message size, processing requirement, etc. Basically, the less frequent the sensor node receives messages, the larger rejection rate, the less processing demand, the smaller message size, we experience less message drops and more choices for system combinations. In general, MIPS or SA-1110 with RSA is a good combination when drop rate is high and ARC3 is the choice when it can handle the messages without any significant drops.

5 Conclusions

We study how to design secure distributed sensor networks with multiple supply voltages to reduce the energy consumption on computation and therefore extend the network's life time. First, we notice that data encryption and decryption is critical for the sensor communication to be secure. However, the energy consumption for encryption and decryption are not the same for most of the public key algorithms. Then we make the observation that the computation requirement of the messages may not proportional to the length of the message. Based on these, we propose to introduce a message header where the message's information is stored. After decrypting the message header, the receiver sensor node will have a better understanding of the entire message without even decrypting the rest of the message. This enables us to make proper selection of supply voltages to reduce the energy consumption. We simulate over a wide range of microprocessors and several different public key algorithm. Simulation results demonstrate the effectiveness of our approach which gives about 60% energy saving despite the overhead of embedding extra information into the message header.

Acknowledgements: This project is in part supported by Minta Martin Research Fund. The authors would like to thank Drs. D.W. Carman, P.S. Kruus, and B.J. Matt from NAI Labs for their help.

Proceedings of the IEEE International Conference on Application-Specific Systems, Architectures, and Processors (ASAP'02)
1063-6862/02 $17.00 © 2002 **IEEE**

References

[1] J.R. Agre, L.P. Clare, G.J. Pottie, and N.P. Romanov, "Development platform for self-organizing wireless sensor networks," *Proceedings of the International Society for Optical Engineering,* pp. 257-268, April 1999.

[2] K. Bult et al. "Low Power Systems for Wireless Microsensors," *Proceedings of the 1996 International Symposium on Low Power Electronics and Design,* pp. 17-22, August 1996.

[3] T.D. Burd, T. Pering, A. Stratakos, and R. Brodersen. "A Dynamic Voltage-Scaled Microprocessor System," *IEEE International Solid-State Circuits Conference,* pp. 294-295, 466, February 2000.

[4] D.W. Carman, P.S. Kruus, and B.J. Matt, "Constraints and Approaches for Distributed Sensor Network Security," *NAI Labs Technical Report #00-10,* September 2000.

[5] A.P. Chandrakasan, S. Sheng, and R.W. Brodersen, "Low-Power CMOS Digital Design," *IEEE Journal of Solid-State Circuits,* Vol.27, No. 4, pp. 473-484, 1992.

[6] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, "Next Century Challenges: Scalable Coordination in Sensor Networks," *ACM/IEEE International Conference on Mobile Computing and Networking,* pp. 263-270, August 1999.

[7] W.R. Heinzelman, A. Sinha, A. Wang, and A.P. Chandrakasan, "Energy-scalable algorithms and protocols for wireless microsensor networks," *IEEE International Conference on Acoustics, Speech, and Signal Processing,* pp. 3722-3725, June 2000.

[8] M. Horowitz. "Low power processor design using self-clocking," *Workshop on Low-power Electronics,* August 1993.

[9] T. Ishihara and H. Yasuura. "Voltage Scheduling Problem for Dynamically Variable Voltage Processors," *International Symposium on Low Power Electronics and Design,* pp. 197-202, August 1998.

[10] V. Von Kaenel, P. Macken, M. G. R. Degrauwe. "A voltage reduction technique for battery-operated systems," *IEEE Journal of Solid-State Circuits,* Vol. 25, No. 5, pp. 1136-1140, October 1990.

[11] J.M. Kahn, R.H. Katz, and K.S.J. Pister, "Next Century Challenges: Mobile Networking for 'Smart Dust'," *ACM/IEEE International Conference on Mobile Computing and Networking,* pp. 271-278, August 1999.

[12] P. Macken, M. Degrauwe, M. Van Paemel, H. Oguey. "A voltage reduction technique for digital systems," *IEEE International Solid-State Circuits Conference Digest of Technical Papers,* pp. 238-239, February 1990.

[13] W. Namgoong, M. Yu, T. Meng. "A high-efficiency variable-voltage CMOS dynamic dc-dc switching regulator," *IEEE International Solid-State Circuits Conference Digest of Technical Papers,* pp. 380-381, February 1997.

[14] T. Pering, T.D. Burd, and R.W. Brodersen. "Voltage Scheduling in the IpARM Microprocessor System," *ISLPED'00: International Symposium on Low Power Electronics and Design,* pp. 96-101, July 2000.

[15] J.M. Rabaey. "Wireless Beyond the Third Generation – Facing the Energy Challenge," *International Symposium on Low Power Electronics and Design,* pp. 1-3, August 2001.

[16] A. Wang, W.R. Heinzelman, and A.P. Chandrakasan, "Energy-scalable protocols for battery-operated microsensor networks," *IEEE Workshop on Signal Processing Systems,* pp. 483-492, October 1999.

[17] F. Yao, A. Demers, S. Shenker. "A Scheduling Model for Reduced CPU Energy," *FOCS'95: IEEE Annual Foundations of Computer Science,* pp. 374-382, 1995.

[18] http://www.janet.ucla.edu/WINS.

[19] http://wins.rsc.rockwell.com/.

[20] http://robotics.eecs.berkeley.edu/~pister/Smart-Dust/.

[21] http://www-mtl.mit.edu/~jimg/project_top.html.

Proceedings of the IEEE International Conference on Application-Specific Systems, Architectures, and Processors (ASAP'02)
1063-6862/02 $17.00 © 2002 **IEEE**