Methodology for Attack on a Java-based PDA

C.H.Gebotys, B.A.White
Department of Electrical and
Computer Engineering
University of Waterloo, Waterloo, Ont
Canada N2L3G1
519-885-1211,
cgebotys@uwaterloo.ca

ABSTRACT
Although mobile Java code is frequently executed on many wireless devices, the susceptibility to electromagnetic (EM) attacks is largely unknown. If analysis of EM waves emanating from the wireless device during a cryptographic computation does leak sufficient information, it may be possible for an attacker to reconstruct the secret key. Possession of the secret cryptographic key would render all future wireless communications insecure and cause further potential problems such as identity theft. Despite the complexities of a Java-based PDA device, this paper proposes and verifies a methodology which confirms EM attacks are possible. The proposed methodology involves pre-characterization of the PDA device through SEMA, thresholding, pattern recognition, and frequency-based DEMA. Results are repeatable over several different secret keys. Unlike previous research the new methodology does not require perfect alignment of EM frames and demonstrates robustness in the presence of a complex embedded system. This research is important for future wireless embedded systems which will increasingly demand higher levels of security.

Categories and Subject Descriptors
C.3 [**Special-Purpose and Application-based Systems**]: smart cards; E.3 [Data Encryption]: Standards (AES);

General Terms
Security, Measurement, Experimentation, Verification.

Keywords
Embedded system

1. INTRODUCTION
As an increasing number of mobile security applications migrate to wireless devices, resistance to attacks on the PDA or cellphone will become a necessity. An attack, if successful, could result in obtaining the secret keys stored in confidential memory in a wireless device. This attack may be possible through loss or theft of the device, or alternatively through temporary access to the device by monitoring the EM waves emanating from the device while performing cryptographic computations. In the latter case the attack may be able to extract the encryption keys, making future wireless communications insecure. These attacks may arise not only from device theft or loss but also during everyday use where unintentional electromagnetic (EM) waves radiated from the wireless device during cryptographic computations may leak confidential data to a nearby attacker.

Although researchers have demonstrated that an EM attack on an 8-bit processor running at 4MHz in a smartcard is viable[3,5], there is limited research on attacks of more complex devices which support mobile code applications. Previous attacks [1,6], have mostly been based upon simple smart card 8-bit or 16-bit processors and in some instances even detailed analysis of cycle by cycle execution of instructions[7] is necessary for the attack. This is largely unlike Java-based systems where just-in-time compilation, dynamic adaptive compilers, java virtual machines, etc, add much more complexity; for example, they have several runtime areas including Java stacks, a heap, a method area, etc. However there is an important need to study EM attacks on Java-based wireless portable devices such as PDAs, cellphones, etc.

2. PREVIOUS RESEARCH
Typically in symmetric encryption the plaintext and key are exclusive or'd together and then indexed into a table, known as the S-box table, as in the table method of the Rijndael advanced encryption standard [2]. The attacker may have control over the plaintext and by guessing each 8-bit secret key value, can partition EM or power frames according to a bit (i.e. bit *'b'* in figure 1) in the guessed data at the output of the S-box table (i.e. a 32-bit value shown as *xxx…b* in figure 1). The trace would be placed into group 0 if the expected value of the bit (*xxx…b[b=0?]*) is zero or alternatively group 1 in the other case (*xxx…b[b=1?]*). By taking the mean of traces in each group and performing the difference of the two means, a differential trace can be created. By recording the magnitude of the differential for each key guess, the attacker can determine the correct key (since it will have the highest differential value). In elliptic curve cryptography (ECC) the data at the output of a double operation can be similarly partitioned according to the guess of the scalar key bit as described in [10].

Although EM attacks on smart cards have been investigated [4,5], EM attacks on other embedded systems have not been widely researched, apart from far field EM emanations from a Palm-Pilot and SSL accelerator[11,12]. Previous research studied

CODES+ISSS'06, October 22–25, 2006, Seoul, Korea.

the correlation of EM variation with data values being manipulated (known as differential EM analysis, DEMA, or DPA for differential power analysis [1]) and instruction sequencing (known as simple EM analysis or SEMA). In the former case, DEMA, the DES encryption[5] was analyzed. Differential EM attacks on embedded low power processors have not been reported in the literature. In most research, good EM or power frame alignment of the attack point is required since most previous differential analyses are performed in the time domain. The exception to this is [9] where the fast fourier transform is calculated, however transformation back into the time domain occurs before differential analysis is performed.

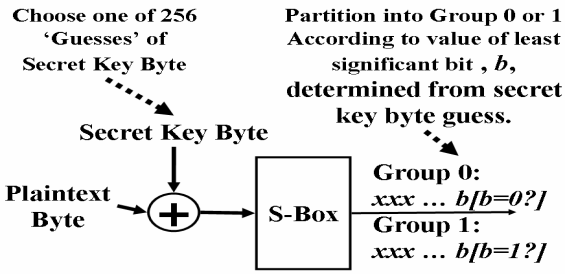


Figure 1. Attack in partial Rijndael AES where partitioning of traces for differential analysis is based on guess of secret key byte.

3. METHODOLOGY

Figure 2 illustrates the proposed new methodology for attack on a Java-based embedded system. It is assumed that the attacker has access to a device, called the characterization device (used in figure 2a)), which is similar to the device under attack (used in figure 2b)). Also the attacker can execute and modify the same encryption algorithm implemented in Java on the characterization device but only execute it on the device under attack.

The first step is to acquire the EM signals from the characterization device while it is executing the cryptographic algorithm (called an EM frame). A SEMA should identify the rounds of the encryption algorithm in step 2 of figure 2a). Next the attacker must be able to find round one and the specific region of attack (i.e. where the Sbox outputs are accessed for the Rijndael encryption algorithm). This may require modification of the Java algorithm (such as truncation of the code followed by SEMA) for verification (see step 4 of figure 2a). The attacker must characterize the delays associated with the acquisition of frames. For example, the attacker will acquire several frames of EM signals which center on the region of attack and further explore a random selection of these frames to determine the time shifts or delays which have occurred. A final acquisition is taken with proper resolution and samples such that most of the frames will contain the region of attack (i.e. the region of attack width plus two times the average delay is typically the frame width). In step 4 the attacker develops a pattern recognition algorithm so that the region of attack is extracted from each EM frame. This algorithm involves EM magnitude thresholding, creating regions of activity, and finally developing a robust set of rules to extract the region of attack. In step 5, the attacker examines various areas within the extracted region of attack in each EM frame. If an area is dynamic, or present in some frames but not others, then it is not a candidate for analysis since it likely represents some dynamic activity in the PDA which is not consistent or associated with the Java code. Areas which are present in all the EM frames (static areas) are candidates for further analysis. In step 6 these static areas are each examined, through extraction and subsequent frequency-based DEMA [13] in step 7.

A successful frequency-based DEMA on the specific area extracted from the region of attack with several sets of acquired EM frames verifies the characterization stage of the methodology. The attack on the device, steps 9 through 12 in figure 2b), can now be performed. This requires SEMA to find and extract region of attack (multi-frame acquisition), subsequent identification and extraction of the area, $Area_k$, identified in figure 2a), and finally frequency-based DEMA to obtain the secret key of the device. This process is repeated for each key byte (*i)* within round 1 until the complete 128-bit encryption key is found. The next section describes the experimental setup which is used for acquiring the EM frames from the device and subsequent experimental section.

3.1 Experimental Setup

A high sample rate oscilloscope, a 1-cm loop EM probe, wide band preamplifier, and a PDA (which was opened to expose the packaged chip over which the probe was placed) were used to acquire EM frames. The oscilloscope has the feature of being able to capture multiple frames, each activated by a trigger signal, until

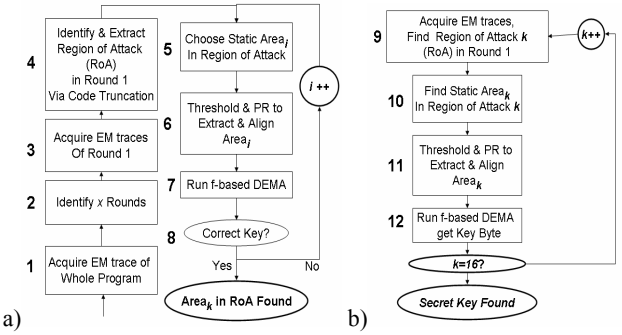


Figure 2. Methodology for characterizing EM a) for attack b)

the scope memory is full. Figure 3 illustrates the experimental setup used to acquire EM signals from the PDA device. The Rijndael encryption algorithm (implemented using the table-based method of [2]) was used to illustrate the EM attack methodology. All encryption algorithms were written in Java and loaded onto the PDA device. A trigger signal was generated from the PDA using the Java code to turn the light emitting diode (LED) on and off. The voltage across the terminals of the LED was used to trigger the scope.

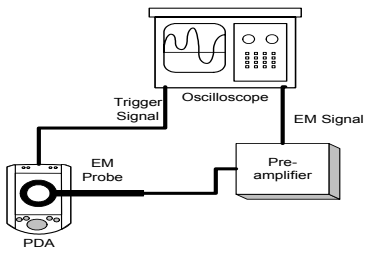


Figure 3. EM experimental setup with probe over PDA device

Figure 4 illustrates simplified Java code of the Rijndael cipher. The plaintext is input to the cipher, and the output is the ciphertext. The cipher uses the table method as described in [2] where 4 S-box tables (*Te0* through *Te3*), each providing 32-bit outputs, are utilized. The beginning of the loop code involves the S-box table accesses. The 32-bit outputs from the S-boxes are exclusive or'd together to create *t[0]* through *t[3]*. The next part of the code involves state creation, where *t0* through *t3* are shifted and transformed into 16 state values, *state[][]*. The last part of the loop code exclusive or's the state with bytes of the next round key, *rkey,* using a routine called *'AddRoundKey'*. This resulting new state is then used to address into the S-box tables of the next loop iteration. Each round is represented by a loop iteration, except the last round which follows the loop. The differential attack on Rijndael focuses on the output of the S-box accesses.

```
public void AES(byte[] plaintext, byte[] ciphertext) {…
for (int round = 1; round < Nr; round++){
t[0]=Te0(state[0][0])^Te1(state[1][1])^Te2(state[2][2])^Te3(state[3][3]);
t[1]=Te0(state[1][0])^Te1(state[2][1])^Te2(state[3][2])^Te3(state[0][3]);
t[2] =Te0(state[2][0])^Te1(state[3][1])^Te2(state[0][2])^Te3(state[1][3]);
t[3] =Te0(state[3][0])^Te1(state[0][1])^Te2(state[1][2])^Te3(state[2][3]);
state[0][0] = (byte) (t[0] >> 24);  state[1][0] = (byte) (t[0] >> 16);
state[2][0] = (byte) (t[0] >> 8);   state[3][0] = (byte) (t[0]);
state[0][1] = (byte) (t[3] >> 16);  state[1][1] = (byte) (t[3] >> 8);
…etc…
state[2][3] = (byte) (t[1] >> 16);  state[3][3] = (byte) (t[1] >> 8);
AddRoundKey(state); }…etc… }
```

Figure 4. Partial and simplified Java Code for Rijndael AES.

The complete Rijndael encryption is executed in a loop on the PDA device for a finite number of iterations using different plaintext input. Specifically only the most significant 8-bits of the 128-bit plaintext are changed in each loop iteration and the other bytes are held fixed (or constant). The value of the most significant 8-bits of plaintext starts at 0 and continues sequentially through to a value of 255 and then repeats from 0. Only the most significant byte of the plaintext changes in order to allow only the first S-box table, *Te0*, in round 1, to change. All other round one S-box table outputs remain constant, hence the noise created by these table accesses is minimized. This greatly simplifies the attack since it can attack the output of *Te0* as well as *t[0]* and the input to Sbox tables in round 2. This approach helped to maximize the probability of a successful attack (since it was not known exactly where the *Te0* load or *t[0]* store/load was located in the EM traces). The plaintexts are then modified and new EM acquisitions are used in order to obtain each key byte (as shown in step 9 of figure 2b)).

4. EXPERIMENTAL RESULTS

The experimental section follows the methodology presented in figure 2 a) characterization and figure 2 b) attack. Finally the information gained through SEMA, truncated Rijndael code analysis, and frequency-based DEMA is then utilized in a full attack on the complete Rijndael code running on the PDA. All results utilize the frequency-based DEMA, since perfect frame alignment is not required. Differential analysis was performed in each case using 512 acquired EM frames. Each EM frame had a length of 100us and contained 50,000 EM samples.

4.1 Characterization of the Device

The first step in characterization of the PDA device (see figure 2), is to run a SEMA on a single frame of EM signals acquired while the entire Java cryptographic algorithm is executed. A single frame of EM data is acquired with a window set wide enough (about 10% wider than the duration of the algorithm) to view the entire cryptographic algorithm. This was approximate since the delay from the software setting of the trigger to the actual trigger signal received by the scope was not known exactly due to LED circuitry and wiring delays. Two scope plots of the acquired EM signals from 10 and 12 rounds of Rijndael executing on the PDA are shown in figures 5a) and 5b). Each of the 10/12 rounds can be seen in the figure (specifically labeled), thus illustrating a SEMA on the device. Since each of the rounds generally utilizes the same instructions (with the exception of the last round), the EM activity is expected to show similar activity in each round. Hence from analysis of the EM trace, it is clear that each round is illustrated with EM signals grouped together with high amplitude separated by low EM amplitude. In each round the high EM signals are expected to be associated with the S-box table accesses and the lower EM activity (shown as separating the rounds) is expected to represent the state creation and add round key operations described in figure 4 of the previous section.

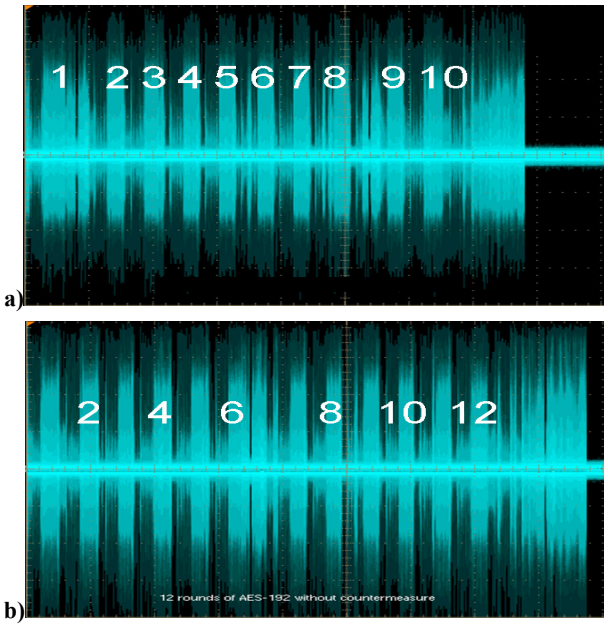
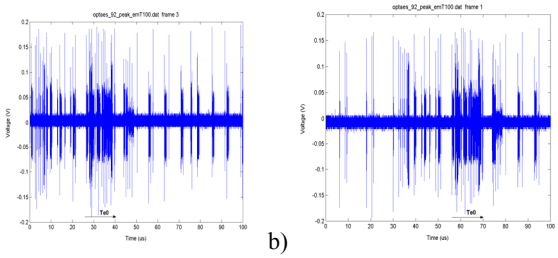


Figure 5. SEMA attack on a PDA device running Java-Rijndael with 10 rounds in a) and 12 rounds in b)

Truncation of the Java code, step 4 of the methodology in figure 2a), was utilized to confirm round 1 and further determine where the first S-box table was accessed. All the rounds in the code were replaced with one line: ***t[0] = Te0(state[0][0]);*** followed by the state assignment code and '*AddRoundKey*' code. Several frames of EM signals were acquired with this truncated Java code. Figure 6a) shows a plot of frame 3 of the EM acquisition of the truncated

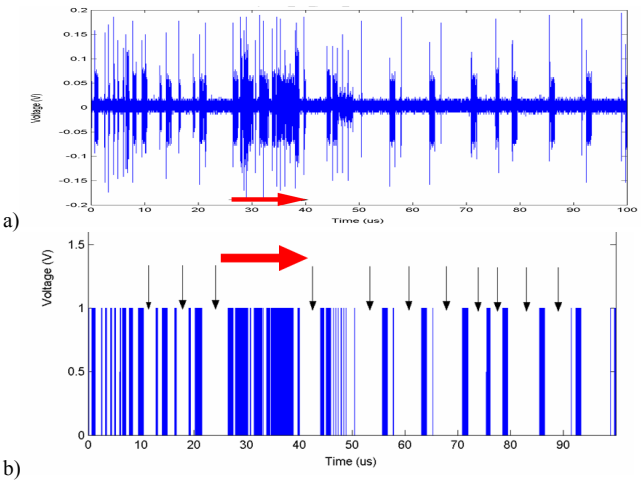
code from step 3 of the methodology. It has been zoomed in to show the *Te0* access region or region of attack. The table access region, labeled as *Te0* in the figure, was identified by comparing this frame plot to another EM acquisition using further truncations of the Java code which omitted the single Te0 table access. Figure 6b) shows another plot of the EM acquisition using frame 1, with the region corresponding to the *Te0* table access again labeled. Figure 6a) and b) further illustrate the severe misalignment (or time shifting) of the frames. The *Te0* region starts 30 us later in frame 1, Fig. 6b), than in frame 3, Fig. 6a). Time shifts of up to 40 us were generally observed in the EM acquisitions.



a) b)

Figure 6. EM acquisition from PDA with truncated AES, capturing 240-340 us, frame 3 on left and frame 1 on right.

Step 4 of the methodology also extracts the region of attack. A pattern recognition algorithm was developed for this task. It extracted and aligned the region of attack from each EM frame. The pattern recognition algorithm consisted of a thresholding step followed by a rule based extraction performed on each EM frame. The thresholding step involved changing each EM frame into a frame of voltages of value zero or one. A threshold level of 0.04V was set since it was just higher than the noise levels. If the voltage exceeded the threshold or was smaller than the negative value of the threshold, then it was replaced by a value of one. Otherwise, if the EM sample fell in-between the positive and negative threshold regions, then it was assigned a value of zero. Figure 7a) shows a plot of EM acquisition frame 3 and the result after thresholding in figure 7b), with the region of attack illustrated with a horizontal arrow.



a)

b)

Figure 7. Frame 3 of acquired EM signals before a) and after thresholding b)

After the thresholding is performed, the region of attack or specifically the *Te0* access region is extracted. In order to extract the region, the thresholded frame is transformed into regions of high activity (dark regions) and regions of low activity (long continuous strings of thresholded zeros). Within the region of interest (*Te0* region) which in this case has high activity, there are mixed samples of zeros and ones. The longest number of continuous zeros within this region is used to set a lower bound on the tolerance parameter. The minimum number of continuous zeros before and after the region of interest sets an upper bound on the tolerance parameter. The tolerance parameter is arbitrarily set to be some value in-between these two bounds. The EM frame of zero and ones is transformed into a series of regions of high activity and low activity using the tolerance parameter. Hence each continuous set of thresholded zeros in the EM frame which is greater than the tolerance parameter (set to 1000 in this case) is kept as a region of zeros, otherwise it is merged into an area of high activity. The small vertical arrows in Fig. 7b) identify these zero or low activity regions. It is important to note that the pattern recognition problem is now reduced from dealing with 50,000 voltages per frame to dealing with 10 regions of low activity.

Pattern recognition tests are next developed to extract the relevant high activity region. In this case the region of interest is identified by using the width of the low activity region immediately before it and the width of the *Te0* access area of interest. In this example, any low activity region which is at least 2000 samples wide and at most 3000 samples wide with a high activity region next to it with a width of at least 5000, is chosen. If more than one region satisfies the criteria, the first region is chosen.

Figure 8 shows the extracted region of attack of frame 1. Step 5 of the methodology examines the various areas ($Area_i$) within each extracted region of attack. Each area is labeled with a letter from *A* to *N* (*i=A to N*), in Figure 8. However not all areas are present in each frame. It was observed that sections which were not present were either replaced with a very short period (close to 0us wide) of low EM activity or were replaced with very low activity for ½ of the original section's typical duration.

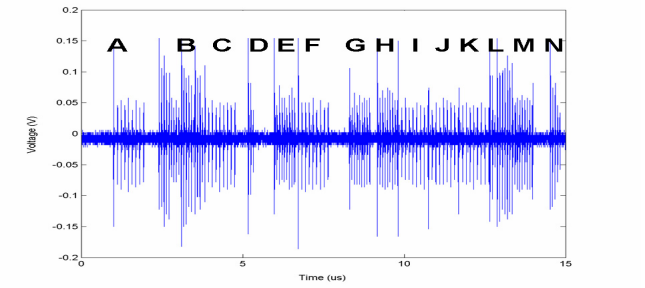


Figure 8. Frame 1 of EM acquisition showing (15us) extracted region of attack with labels A to N.

Section *N* was chosen as the first area to analyze since it was present in each frame and appeared towards the end of the frame (hence was candidate for representing the table value being accessed). Section *N* was extracted, in step 6 of the methodology in figure 2a), by taking samples immediately before the end of the region of attack and 100 samples following the end of the region of attack as well. This processing created only 300 extracted samples per frame, as compared to the 50,000 voltage samples in

the original frames. Figure 9 shows the extracted section *N* which consists of 5 peaks of reducing EM amplitude. Analysis of other sections which appeared in all EM frames (such as section *L*) were also performed, but did not reveal the correct key.

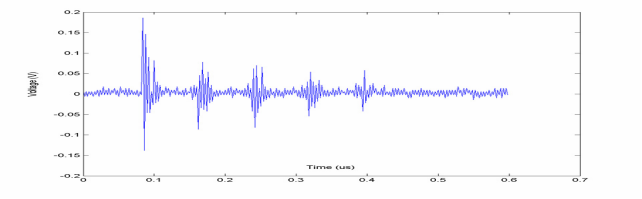


Figure 9. Frame 3 of EM acquisition showing (0.6 us section *N*) $Area_k$ extracted from region of attack

The frequency-based DEMA, step 7 of the methodology, was run on the extracted section *N*. It successfully found the correct key. New EM frames were acquired with several different keys to confirm the methodology. It is interesting to note that time-based DEMA analysis was not successful in finding the correct key. Further investigation showed that although the extracted section *N* samples were almost synchronized, there were still a few ns of remaining time shifts.

In summary, based upon analysis of the truncated code, section *N* was determined to represent the attack area. The next section will utilize the full AES Rijndael encryption code to illustrate a complete attack as outlined in figure 2b).

4.2 Attack of the Device

Since the previous section analyzed the device under attack to determine that section *N* represents the attack point, the full AES code will next be attacked using this information. This section describes the use of the full (not truncated) AES encryption Java code. Thresholding and pattern recognition are both used again to extract the appropriate regions for analysis.

Figure 10a) shows a plot of frame 4 of the EM acquisition (using key 92 decimal). Four regions that look similar to the one *Te0* access region for the PDA truncated AES are labelled with red arrows pointing up and will be referred to as Group 1 through 4 from left to right. These four regions of attack corresponded to the four S-box table accesses defined in the third line of Java code in figure 4. From the SEMA performed, it was observed that there were four groups, each consisting of 4 S-box table accesses. Each group corresponded to processing of the table accesses, specifically *Te0, Te1, Te2, Te3* from figure 4. Section *N* consistently only appeared in every 4th group. For example figure 11 illustrates Group 1 (the first red arrow on the left in figure 10a)) of the first candidate region of attack, where there is no section *N*. Section *N*, which only appeared in Group 4, was used again as the pattern recognition target for isolation and extraction.

The next step, step 11, in the attack methodology was to perform thresholding and pattern recognition in order to extract section *N*. The thresholding step was the same as described in the last section. Figure 10b) illustrates the thresholded result corresponding to the original EM acquisition frame 4, shown in figure 10a). For creating the regions of high and low activities, the tolerance parameter is set to be greater than any gaps (strings of zeros) within the Group 1-4 regions but less than the width of the strings of zeros preceding and following the Group 1-4 regions. Again the pattern recognition problem is reduced from dealing with 50,000 voltages per frame to dealing with 5 regions of high activity.

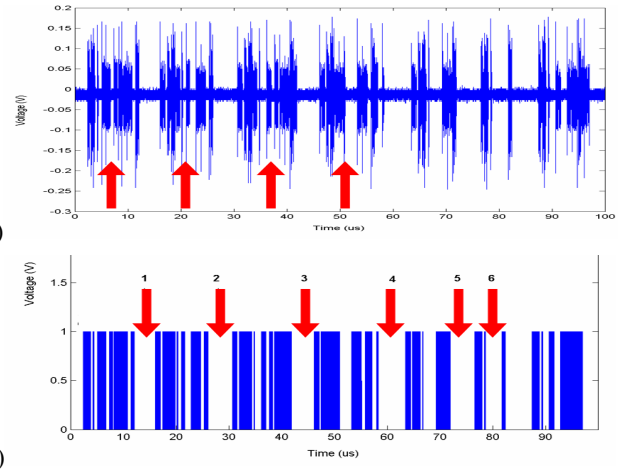


Figure 10. a) before thresholding and after thresholding b)

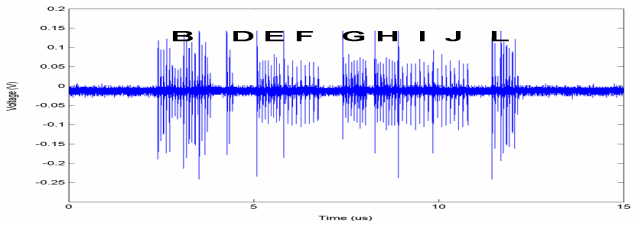


Figure 11. EM acquisition frame 4 for PDA full AES, showing 15 us Group 1 with no section N

The arrows in figure 10b) identify the long strings of thresholded zeros. Again the pattern recognition was designed to search for a width (between 2000 and 3000 samples in this case) and separation pattern (separation of greater than 5000 samples) which identifies the thresholded zeros region preceding the Group 4 region.

Using the EM acquisition section *N* was extracted from each frame. The extraction took 200 samples (0.4 us) from the end of the Group 4 region (section *N*) and 100 samples (0.2 us) following the end of the Group 4 region. There were now only 300 extracted samples per frame. The pattern recognition and extraction of Group 4 section *N* was extremely robust. The number of frames where pattern recognition failed was at most 2 for each of 8 EM acquisitions using 4 different keys. Mostly this failure is a result of the first acquired EM frame which is always "abnormal", not having the usual pattern of groups and sections.

The frequency-based DEMA was run on the extracted Group 4 section *N* for each frame. It was robust and successfully found all correct key bytes for each of 8 EM acquisition acquisitions using 4 different keys. The next highest magnitude from the frequency-based DEMA results was only 28-44% of the correct key magnitude. Figure 12 shows detailed plots of the all keys search utilizing the frequency-based DEMA for 4 different keys.

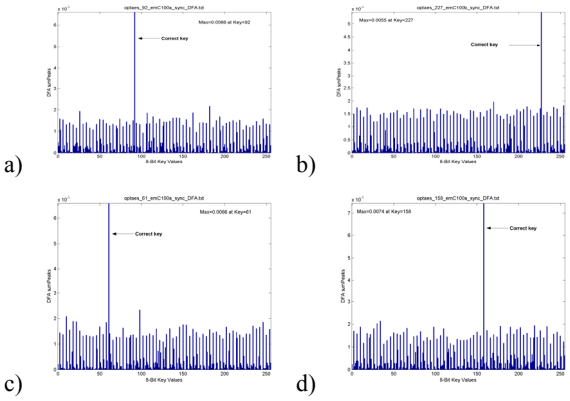


Figure 12. All keys search using frequency-based DEMA results, for keys 92, 227, 61, 158 in a)-d).

5. DISCUSSION AND CONCLUSIONS

In summary this paper presents a new methodology for attacking a Java based cryptographic algorithm which is demonstrated using EM signals obtained from a wireless PDA device. The methodology together with frequency-based DEMA, created a successful side channel attack on a real PDA running Java code. The attack is robust and works for all keys which were tried. The results demonstrate that embedded mobile code must contain countermeasures in order to avoid attacks where the secret cipher keys can be extracted.

It is highly likely that section *N* used in the attack was the store of value *t[0]*. This is unlike previous research which had not considered attacks using *t[0]*. Only the S-box tables were considered viable attack points [7]. Hence although the S-box tables were not directly attacked as in most previous research [7], an attack was still successful and viable if the attacker has control over the plaintexts. It is also interesting to note that further analysis did not successfully attack the S-box table accesses. It may be that the S-box tables were loaded all at once into cache earlier in the code, and hence more difficult to find.

The frequency-based DEMA was very successful once the area of attack was extracted. However when larger windows were used, such as the entire *Te0* access region including section *N,* the attack failed. This is likely caused by the frequency content of the voltage transitions in the other included sections obscuring the section *N* frequency content. Hence frequency-based DEMA still requires some isolation and extraction of signals. However it requires far less work than time-based DEMA which was not successful in obtaining the correct key on the PDA. The time-based DEMA requires excellent EM frame alignment in order to work. Excellent EM frame alignment may be difficult to achieve without complex automatic pattern recognition or other means.

The thresholding and pattern recognition may need to be adjusted for EM signals from different types of devices or different implementations of the Java code. The algorithms for thresholding and pattern recognition, providing extraction of regions in the EM, are relatively simple. This is due to the Java execution, where the EM activities are often separated by low EM activity (such as the case for section *N*) and also due to the frequency-based DEMA which does not require perfect alignment. It was assumed in this paper that the cipher implementation is the same in both the characterization device and the device under attack. However it may be possible for the attacker to explore various implementations on the characterization device and determine which implementation more closely matches the EM signals from the device under attack.

The proposed methodology may also be applied to power signals, however these are more difficult to obtain from the chip of the PDA housing the respective processor. Power signals obtained from the battery of the mobile device may also be difficult since signals likely would be buried amidst the current drain of other components requiring power. For example on PDA devices, there are many different components including power management circuitry, radio circuitry, baseband processor, etc.

For the first time using real EM measurements from a PDA device executing Java-based cryptography, a methodology for attack is proposed and verified. The results indicate that the methodology provides a robust approach for finding the cryptographic key. Unlike previous research, a methodology based on thresholding and pattern recognition greatly reduces the complexity of the EM analysis. Results show that a Java-based cipher provides many more opportunities for attack since many stores and loads to and from memory are performed due to the Java virtual machine. This research demonstrates that countermeasures must be employed to protect all attack parts of an algorithm. This research is crucial for supporting low energy security for embedded systems which will be prevalent in wireless embedded devices of the future. This research is supported in part by grants from NSERC and OCE.

6. REFERENCES

1 P.Kocher, J.Jaffe, B.Jun "Differential Power Analysis" Crypto'99, LNCS 1666 (1999)388-397

2. Dr.Brian Gladman, "A Specification for Rijndael, the AES Algorithm", at fp . gladman . plus . com / cryptography_ technology / rijndael / aes.spec.311.pdf (2003)

3. D.Agrawal et al. "The EM side-channel(s)" CHES 2002 (2002) 29-45

4. K.Gandolfi etal. "Electromagnetic Analysis: concrete results" CHES 2001, LNCS 2162, (2001) 251-261

5. D.Agrawal, etal. "The EM side-channel…methodologies" at http ://www.research.ibm.com/intsec/emf.html

6. M.Akkar, etal. "Power analysis, what is now possible…", LNCS 1976 (2000) 489-502.

7. K.Itoh etal. "DPA countermeasure based on the masking method", LNCS 2288 (2002) 440-456

8. T.Messerges "Using 2nd order power analysis to attack DPA resistant software", LNCS 1965 (2000) 238-251

9. J.Waddle, D.Wagner "Towards efficient second-order power analysis" CHES 2004, LNCS 3156, (2004) 1-15

10. J. Coron, "Resistance against differential power analysis for elliptic curve cryptosystems", *CHES 1999*, LNCS 1717 (1999) 292-302

11. D.Agrawal, etal. "Advances in Side-Channel Cryptanalysis EM analysis and template attacks" RSA Cryptobytes, Vol6 No1 (2003) 20-32

12. D.Agrawal, etal "Power, EM and all that: is your crypto device really secure?" presentation ECC workshop http://www.cacr.math.uwaterloo.ca/conferences/2003/ecc2003/ rohatgi.ppt (2003)

13. C.Gebotys, S.Ho, A.Tiu "EM Analysis of Rijndael and ECC on a Wireless Java-based PDA" Proceedings of CHES 2005, LNCS 3659, Springer-Verlag GmbH, pp.250-265.