The Temperature Side Channel and Heating Fault Attacks

Michael Hutter[1] and Jörn-Marc Schmidt[2,*]

[1] Institute for Applied Information Processing and Communciations (IAIK),
Graz University of Technology, Inffeldgasse 16a, 8010 Graz, Austria
Michael.Hutter@iaik.tugraz.at
[2] secunet Security Networks AG,
Mergenthalerallee 77, 65760 Eschborn, Germany
joern-marc.schmidt@secunet.com

Abstract. In this paper, we present practical results of data leakages of CMOS devices via the temperature side channel—a side channel that has been widely cited in literature but not well characterized yet. We investigate the leakage of processed data by passively measuring the dissipated heat of the devices. The temperature leakage is thereby linearly correlated with the power leakage model but is limited by the physical properties of thermal conductivity and capacitance. We further present heating faults by operating the devices beyond their specified temperature ratings. The efficiency of this kind of attack is shown by a practical attack on an RSA implementation. Finally, we introduce data remanence attacks on AVR microcontrollers that exploit the Negative Bias Temperature Instability (NBTI) property of internal SRAM cells. We show how to recover parts of the internal memory and present first results on an ATmega162. The work encourages the awareness of temperature-based attacks that are known for years now but not well described in literature. It also serves as a starting point for further research investigations.

Keywords: Temperature, Side Channels, Fault Injection, Negative Bias Temperature Instability, AVR, Smart Cards

1 Introduction

It has been known since the late 1990s that implementations of cryptographic algorithms leak information from different side channels. The first paper that demonstrates the exploitation of a side channel was published by P. Kocher [24] in 1996. He highlighted that implementations might provide timing characteristics that leak information of private keys. Attacks are therefore able to extract the keys by simply measuring the runtime of the implemented algorithm. Three years later, he introduced the power-consumption side channel together with B. Jun and J. Jaffe in [25]. They observed that key material can be also extracted from the power consumption of cryptographic devices. Since then, many researchers started to investigate the properties of these leakages on different

[*] This work was done while the author was with Graz University of Technology.

platforms and devices. They also proposed to exploit other powerful side channels like electromagnetic (EM) emanation [1,13,16,31]. Up to now, the power and the EM side-channels have been widely established and used in academia and industry precisely because of simplicity, low-cost, and efficiency compared to other existing side channels.

Other lesser known and rather more exotic side channels (but not necessarily less powerful) are, for example, acoustic or optical emissions. Acoustic side channels have been first introduced by A. Shamir and E. Tromer [39] in 2004. They extended their work recently in 2013 [17] and provided a wide range of possible acoustic attacks, e.g., on GnuPG's RSA implementation. Heat causes mechanical stress which produces acoustic noise. This noise contains information about the power usage of CPUs and thus information about the processed data. A very related attack was also presented by D. Asonov and R. Agrawal [4] who exploited the fact that PC keyboards emanate different sounds that can be recognized at a distance. Improvements of the latter attack were reported by L. Zhuan et al. [43] in 2009. Optical emissions, in contrast, were investigated by J. Ferrigno and M. Hlaváĉ [15] as well as A. Schlösser et al. [35] who targeted an AES implementation. S. Skorobogatov [38] used a low-cost CCD camera to analyze the leakage of photons emitted from SRAM, EEPROM, and Flash memories.

The temperature side-channel has been often cited in literature, see for example [6,8,10,11,22,23,30,36,42]. However, most of the publications only mention the existence and possibility to exploit this channel but without providing further investigations. In particular, H. Bar-El stated in [6] that temperature attacks on smart cards are "never documented in the open literature to the author's knowledge". The only publication that pinpoints the existence of the temperature side-channel is due to J. Brouchier et al. [10,11] from 2009. They showed that a cooling fan can carry information about the processed data indirectly through the dissipated temperature of a CPU. Within an experiment, they demonstrated how to extract bits from a secret password or possible RSA key (by assuming a low-frequency leak of the bits though, i.e., leaking a bit per three minutes). Furthermore, they emphasized that IP cores integrated in FPGAs might leak information to other IP cores in the system via the temperature side channel.

There are a few more papers on *active* temperature attacks, i.e., attacks that actively tamper the environmental temperature of a device (cooling or heating). Most of them demonstrate the efficiency of low-temperature attacks, e.g., as reported by S. Skorobogatov [40] and D. Samyde et al. [34] in 2002. They showed that by cooling down SRAM devices up to -50 °C, they were able to *freeze* the data and to recover the content of the memory even after seconds after power-down (by exploiting the data retention property of SRAM cells). The same idea was used by T. Müller et al. [29] who presented a tool called FROST[3]. The tool is able to recover the RAM content of modern Android smart phones similar to *cold boot attacks* on PCs [21]. High-temperature attacks, in contrast, have been investigated by J.-J. Quisquater and D. Samyde [32] who observed memory errors after hours of extensive heating. Similar results have been reported by

[3] FROST stands for *Forensic Recovery of Scrambled Telephones.*

S. Govindavajhala and A. Appel [19] who were able to induce errors into memories using a 50 watt spotlight clip-on lamp. By heating an IBM JVM to 100 °C, they were able to inject faults with a probability of 71.4 % before their machine crashed.

In this paper, we describe a set of temperature-related attacks on common AVR and PIC 8-bit microcontrollers. There are three main contributions listed in the following:

1. We first characterize the "temperature side-channel" by presenting results of data leakages of AVR and PIC microcontrollers. We investigate the leakages and identify the linear relationship between heat radiation and circuit activity. It shows that the analyzed devices leak the Hamming weight of the processed data via the (low-frequency) temperature side channel.
2. We conduct high-temperature fault attacks on AVRs by operating the devices beyond their specified temperature ratings (>150 °C). A practical attack is shown on an RSA implementation where we successfully extracted the used private key.
3. Finally, we exploit the physical property of data remanence attacks on AVRs. By extensive heating, constant data like the private key gets burned in memory that can be recovered even after years. We identify permanent as well as transient NBTI degradation components and were able to fully recover 65 % of the entire memory of an ATmega162.

The rest of the paper is structured as follows. In Section 2, we characterize the temperature leakage of AVR microcontrollers. Section 3 presents results of heating fault attacks on RSA. In Section 4, we describe data remanence attacks. A discussion of the results is given in Section 5.

2 Temperature Leakage Characterization

In this section, we aim to characterize the temperature side channel by analyzing the leakage of an 8-bit ATmega162 AVR microcontroller [5]. This family of microcontrollers is widely used in embedded systems such as industrial automation, control, or in smart cards, e.g., integrated in the Funcard, ATmega Card, M2, KNOT, and Titanium programmable smart cards. First, we describe the setup to measure the temperature dissipation of these devices. Afterwards, we characterize the side channel in terms of its physical limits and possible exploitation efficiency.

2.1 The Setup to Measure the Temperature

Our setup is very similar to setups that are used to perform power-analysis attacks. Instead of measuring the power consumption of our target device, we measure the dissipating temperature using a PT100 sensor element. The PT100 is a very common thermometer applied in various industry products. It measures

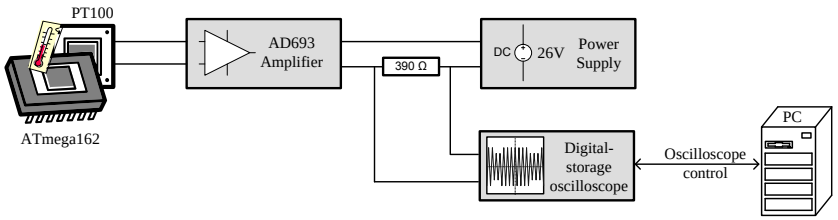

Fig. 1: Schematic view of the used setup to exploit the temperature side channel.

the resistance of a platinum element having a resistance of 100 Ohm at 0 °C. Next to the PT100, we use an AD693 amplifier (voltage to current converter) that provides a pre-calibrated Resistance Temperature Detector (RTD) interface allowing accurate measurements in the temperature range of 0 to +104 °C. In that configuration the output current span is 4 to 20 mA which was measured by calculating the voltage drop over a 390 Ohm resistor in series to the power supply. We used a standard 1 GHz digital oscilloscope for that purpose and connected it to a PC that runs Matlab for controlling the measurement process. Figure 1 shows the schematic of the used setup.

In order to allow an accurate characterization of the temperature leakage, we decided to decapsulate the chip from the rear side and to measure the temperature dissipation directly on the surface of the silicon substrate. Figure 2 shows the decapsulated ATmega162 and the PT100 touching the rear side of the chip die. The PT100 has been surrounded by a thermal-conductance paste $(4 - 10\,W/(m \cdot K))$ to allow a stable and accurate sensing of the temperature. Note that for the targeted device we had to remove the copper plate which is located below the chip die and the plastic package as similarly done by [36,41]. Subsequently, we polished the substrate but we did not apply any additional thinning procedures. This can be simply done by using a skew driver.

In general, silicon substrate has a good thermal conductivity which is much higher than the conductivity of the surrounding die package. So it is advantageous to decapsulate the chip but this is not necessarily required. We also performed the experiments without decapsulating the chip and measured the temperature on the surface of the package (which corresponds to a non-invasive attack). We observed that the leakage is slightly weaker but strong enough to obtain similar results. Figure 3 shows a picture of the overall setup including a controller board that is used to communicate with the ATmega162 over a serial-communication interface.

2.2 Temperature Analysis

In order to characterize the leakage of our targeted device, we measured the temperature dissipation of various processed intermediate bytes and used a long acquisition window to evaluate the impact of thermal conductivity and capacitance. As a target operation, we made use of a `MOV` instruction and moved all

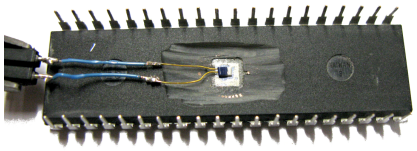

Fig. 2: Rear-side decapsulation of an ATmega162 to provide direct contact to the silicon substrate for the PT100.

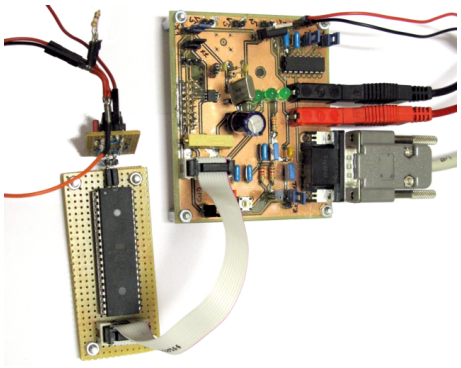

Fig. 3: Side-channel measurement setup including controller board, ATmega162, and temperature-sensing circuit.

possible values of one input byte (i.e., 256) to 24 internal registers (the remaining 8 registers are used for loop indices and other temporary data). These `MOV` instructions were executed in a loop where the loop index and execution duration had been configurable by the PC. The temperature dissipation was then measured for a period of 20 seconds where in the first 10 seconds only zero values were moved to the registers and in the last 10 seconds, the current input-byte value was written to all registers[4]. For each input-byte value, we measured 100 traces and averaged them to reduce noise. Figure 4 zooms into the acquisition window showing the most interesting 10 seconds during the transient phase. It shows that in the first part of the traces, the temperature is equal for all inputs (the temperature is decreasing because of the fast acquisition runs and the higher temperature dissipation of previously measured traces). At the point when the actual input byte is written, the temperature increases depending on the Hamming weight of the processed value.

Figure 5 shows the temperature dissipation of all input-byte values at the time right before the end of the acquisition window, i.e., after about 18 seconds. It clearly shows that the temperature corresponds to the Hamming weight of the processed intermediate values ranging from 26.6 to about 26.8 °C. The temperature therefore linearly correlates with the power model of the device which was known and characterized using power analysis. If the circuit activity is high (meaning if there occur many bit transitions), the dissipated temperature increases. If the circuit activity is low due to less or no bit transitions, the temperature decreases. This, however, holds true not only for dynamic power consumption (caused by charging and discharging capacitances) but also for static power consumption (caused through sub-threshold and leakage current). Both components cause an averaged DC increase in temperature.

[4] We set all registers to zero before writing of new values to guarantee the transitions of all bits (avoiding Hamming-distance leaks).

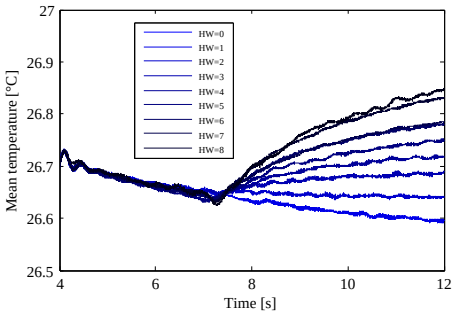

Fig. 4: Slow temperature increase of all Hamming weights that are processed by the ATmega162.

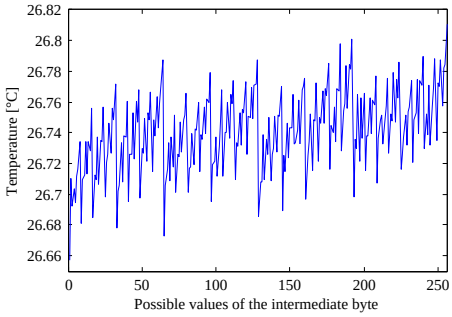

Fig. 5: The ATmega162 leaks the Hamming weight of all 256 possible intermediate values through the temperature.

From Figure 4 it is also observable that the temperature increases and decreases very slowly. While the circuit activity is constantly high or low, the temperature increases or decreases by about 0.3 °C over a period of 10 seconds. This has its reason in the following facts. First, the temperature variation is limited by the physical property of thermal conductivity. The heat flow from the die (causing thermal power) to the sensing element can be seen as an RC network including resistance and capacitances for the junction (i.e., chip die), the package/case, heat sink, and ambient air. This RC network has the typical property that it consists of large (thermal) capacitances that causes the network to behave like a low-pass filter. The cut-off frequency is thereby very low (typically between some Hz and tens of kHz [2]). This means also that high frequency leakages (> 1 MHz), which usually appear in CMOS devices, will not be easily exploitable from that side channel because the information will be largely filtered by the RC network. The measured temperature signal at the sensor element therefore contains all superimposed and integrated signal components of the power consumption. As a second reason, the used temperature sensors have a certain response time and acquisition resolution. In our experiment, we used a PT100 that has a thermal response time of 100 milliseconds and a resolution of 0.01 °C.

It shows that the temperature side-channel has a very low bandwidth limiting practical attacks. In the following, we discuss possible attack scenarios for low-frequency temperature leakages:

1. An attack exploiting the temperature side-channel is possible in case the leakage of the data is present over a period of several milliseconds or seconds. If an application repeatedly checks a password, for example in a loop, enough information is available even in limited frequency bands that allows low-bandwidth attacks [10,11].
2. Many RSA implementations involve operations that take a long time, e.g., modular exponentiations. These operations create signals in a low frequency band that can be revealed by low-bandwidth acoustic attacks as recently

shown by D. Genkin, A. Shamir, and E. Tromer [17]. These low-bandwidth signals can be also extracted from the temperature side channel.

3. A very powerful attack which is not well investigated yet is the exploitation of static power-consumption leakages. Most of side-channel based analyses are exploiting the dynamic power-consumption which is the main contributor to the total power consumption of electronic devices. With shrinking CMOS technology, static leakages become more significant. Temperature attacks exploiting the static power-consumption benefit from less strict timing constraints because the leakage is *statically* available over an infinite period of time. A. Moradi recently demonstrated successful power-analysis attacks exploiting the static leakage of FPGAs in [28]. Other works also characterized and exploited static leakages of CMOS devices, for example, in [18,26].

3 Exploiting Heating Faults on AVRs

In this section, we intentionally operate a target device beyond its maximum temperature ratings in order to produce exploitable faults due to extensive heating. Each electronic device specifies a certain temperature range where the correct operation is guaranteed by the manufacturer. If these limits are exceeded by external influences, data might get modified that is stored in memories or processed by the CPU. Faulty cryptographic operations can then be exploited in attacks to reveal the secret key [6,7,23,36].

In the following experiments, we used the same ATmega162 as used in the previous experiments. To prove the practicability of our attack, we implemented RSA, induced heating faults, and successfully extracted the private key used during encryption of data, cf. Bellcore attack [9].

Setup and RSA Implementation. We used a low-cost laboratory heating plate from Schott instruments (SLK 1) to heat-up and induce faults in an ATmega162. The microcontroller has been placed directly on top of the hot-plate surface, lying top-side down to allow a good heat transfer. The temperature of the internal IC has then been measured by calculating the mean of two PT100 sensors to be more accurate. One PT100 has been placed on the rear side of the ATmega162, the other PT100 has been placed directly on the surface of the heating plate. Both PT100 are connected to an oscilloscope similar to the setup described in the previous section. Figure 6 shows the setup.

We connected and used only six mandatory pins of the ATmega162: power supply (VCC and ground), serial communication (RX and TX), clock signal, and reset. For these connections, we used exposed wires to avoid any contact to the hot plate and the melting of solder[5] during long-time heat exposure. As a controlling device, we used an FPGA board (Spartan-3) that is connected with the measurement PC.

[5] The temperature melting point of Sn63/Pb37 lead solder, which is commonly used for electrical soldering, is 456 K (183 °C).

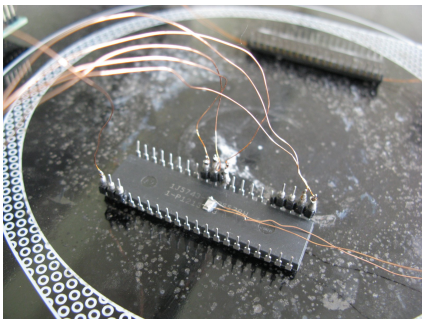

Fig. 6: Heating plate with two PT100 sensors measuring the rear-side and front-side temperature of an ATmega162.

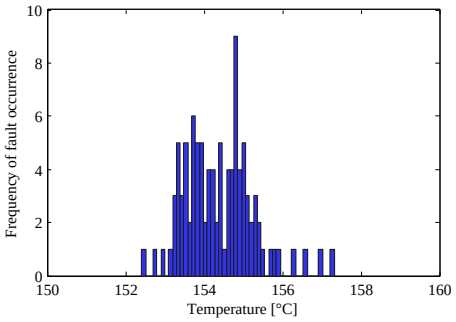

Fig. 7: Distribution of fault occurrence between 150 and 160 °C. Mean fault-induction temperature is 154.4 °C.

We decided to target an RSA implementation that implements the Chinese Remainder Theorem (CRT). This attack is very simple and well documented in literature [6,9,36]. Only one single fault during the computation can reveal the secret RSA primes p and q. The attack and evaluation of faulty computations has been performed using Sage [33]. A description of the attack is given in Appendix A.1.

3.1 Heating-up the Target

The used heating plate provides ten heating stages that can be adjusted manually going up to 1 000 K, i.e., about 727 °C. By manual adjustment, we were able to heat up the device under test up to 150 °C within about 5 minutes.

In a first observation, we identified that the ATmega162 does not respond to requests anymore if the heating temperature is higher than around 160 °C. Note that this is much higher than the *operating temperature* maximum ratings given in the device specification [5], i.e., between -55 °C and +125 °C and it is approximately as high as the given maximum *storage temperature* rating of the device which is 150 °C. As a key observation, we identified that the device starts producing faults between a certain heating window of about 152 and 158 °C. During this window, the probability is high that the device outputs an incorrect result because of an induced fault during the computation of the implemented algorithm.

In order to quantify this behavior, we performed several measurements on that device taking about 70 minutes. We performed an RSA decryption operation every 650 milliseconds and kept the heating temperature between 150 and 160 °C by manually adjusting the heating plate. As a result, we got 100 faults where 31 of this set have been exploitable, i.e., the attack revealed one of the used prime moduli. In the other cases, the fault was induced during I/O communication or other parts of the computation such that the difference of the faulty and a correct signature output was coprime to the RSA modulus and

thus was not factorizable. In addition to these outcomes, we identified that 16 out of the 31 faulty computations, revealed the prime modulus p and 15 revealed the prime modulus q. Hence, the probability that a fault reveals p or q is expected to be about 50 %—a result that was expected since both p and q have the nearly the same size and the modular exponentiation with these primes take the same amount of time and therefore provide the same fault-induction window. Furthermore, 23 out of the 31 faulty computations have been unique and different whereas 7 faults have been repeated, i.e., the RSA decryption yielded an output that was already obtained in a previous measurement. For the latter case, we can therefore assume that either some internal memory locations or internal logic parts are more sensitive to heating than others, thus causing the same RSA outputs. Figure 7 shows the frequency of the induced faults. Most of the faults occurred between 152 and 158 °C having a mean fault-induction temperature of about 154.4 °C.

We made the same experiment also using other ATmega162 devices (new once) in order to verify our results. First, it showed that the mean fault-induction temperature for each device is slightly different and varies per device. Second, it showed that the number of faults is different per device (e.g., we got 182 faults within 30 minutes for another device) which is likely because the temperature has to be adjusted manually in our setup and varies per measurement. But the attack succeeded for all devices within less than 30 minutes.

4 Data Remanence Attacks on AVRs

In this section, we characterize the property of data remanence effects of the internal SRAM cells of an ATmega162. It is known that data which is stored in the same location after each power-up of a device (such as a secret key that is loaded from program memory/flash to RAM) leaves a permanent mark that can be recovered as, for example, detailed by P. Gutmann in [20]. He observed that data that is stored in SRAM or DRAM for a long period of time remembers the value when powered-up again even after years. This effect has been practically exploited in an attack by R. Anderson and M. Kuhn [3] who were able to recover $90 - 95$ % of a DES master key used by an old bank security module from the late 1980s. C. Cakir et al. [12] have characterized the data remanence effect on newer 65 nm CMOS RAMs which was considered to be not that efficient because of the newer SRAM structures. However, they were able to recover about 18 % of the entire SRAM content (in fact, 82 % have been recovered correctly out of 22 % predictable bits).

The SRAM data remanence effect can be explained as follows. SRAM cells that are exposed to extensive heating are subject to accelerated aging where internal transistor parameters get changed. This effect is known under Negative Bias Temperature Instability (NBTI) and has been first observed in the late 1960s. Since then, many researchers identified that this effect decreases parameters such as speed, drive current, and noise margins. In fact, NBTI occurs when transistors are stressed with negative gate voltages at elevated temperatures,

e.g., during burn-in stress. Then, the (absolute) threshold voltages of the transistors increase which change the preferred power-up state of SRAM cells. Thus, transistors get "weaker" and tend to a certain bit value after power-up. Note that NBTI has been basically observed for both PMOS and NMOS transistors while PMOS transistors are more affected [37].

The study of M. Ershov et al. [14] showed that there exist two NBTI degradation components: one component that remains after burn-in stress (permanent damage), and another component (transient damage) that recovers within a certain amount of time (seconds up to days). In the following, we identify these two degradation components in practical experiments on the ATmega162. We used the same setup as it was used in the previous section to heat up the device.

Before we started the experiments, we determined the preferred power-up values of the ATmega162 in order to evaluate the effect of burn-in stress. A small program was written that allows reading and writing of SRAM content over the serial interface. A byte array of 6 144 bits (out of the available 8 192 bits) was used for testing. After we disconnected all I/O connections from the device for some milliseconds[6], which has been accomplished with our flexible FPGA controller board, we read out the SRAM content. This has been done 100 times to average noise. It showed that 3 101 bits (50.47 %) are powered-up to the value *one* and 3 043 bits (49.53 %) are set to *zero* on average, i.e., there was almost no bias in the distributions.

4.1 Permanent Data Remanence Effects after Burn-In Stress

We programmed the internal SRAM of the ATmega162 with normally distributed random data. We set 3 072 bits to 0 and 3 072 bits to 1 at random locations[7]. A first test after a power-up reset showed that the probability of guessing the bits correctly was around 50 % as expected. After this test, we exposed the device to extensive burn-in stress to accelerate aging effects. The stress conditions were a high ambient temperature of about 100 °C and an over-voltage supply of 5.5 Volts. We applied the stress over a period of 36 hours and read out the power-up SRAM content afterwards. The read out values are then compared with the initial values.

It showed that the number of bits that are one or zero got biased due to NBTI degradation. There were 3 210 bits (52.24 %) set to one and 2 934 bits (47.75 %) set to zero after the burn-in stress. Furthermore, 919 bits of the total memory changed their state (i.e., 15 %): 405 bits moved from 1 to 0 and 514 bits moved from 0 to 1. From these 405 bits there were 393 bits that moved to the correct value zero (97.04 %) and 489 from 514 bits moved to the correct value one (95.14 %).

[6] We disconnected not only the power supply but also the RS232 interface and the clock signal to guarantee that the device (and SRAM respectively) is completely unconnected and not powered by I/O interfaces. Note also that we used hardware relays to actually disconnect all connections.

[7] We do not assume the knowledge of "preferred power-up values" before burn-in stress to guarantee a realistic attacking scenario.

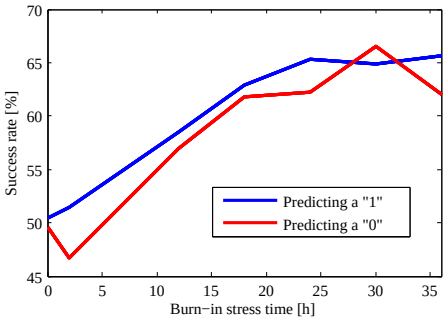

Fig. 8: Probability of predicting a SRAM bit correctly increases from 50 % to about 65 % after burn-in stress.

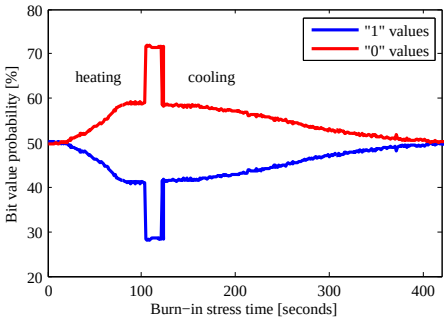

Fig. 9: Bit-value distribution during heating and cooling (transient data remanence effect).

Note that in our analysis, we were only able to identify half of the *unstable* SRAM cells. This is because from the set of unstable SRAM cells, statistically only half of the bits changed to the correct value while the other half already held the correct value. So, in our experiment, we can assume about 30 % of unstable SRAM cells and 70 % of cells that are *stable*, i.e., they did not change at least during our burn-in stress of 36 hours. Now, by guessing 50 % of all stable cells correctly and by assuming a high probability of guessing all unstable bits correctly, we are able to successfully predict 65 % of the entire SRAM memory. This nicely corresponds to our practical results where we achieved a success rate of about 63 %, i.e., 3 842 bits have been predicted correctly as shown in Figure 8. It shows that there is a huge increase during the first 20 hours. After that, the probability keeps nearly constant. We also performed a measurement one week after the burn-in stress and identified no changes in the success rate.

4.2 Transient Data Remanence Effects during Burn-In Stress

We also characterized the transient data remanence effect during the burn-in stress process. For this experiment, we used a new ATmega162 and read out the SRAM content every 4 seconds while the device was heated up. Figure 9 shows the bit-value distribution during the burn-in stress and right after it. In the first 115 seconds, the device is heated up to 170 °C. After that, the heating plate was turned off to cool down the device. Between 105 and 125 seconds, the device was overheated which produced a significant jump in the probability distribution.

The experiment showed that the effect of NBTI degradation of PMOS and NMOS transistors in SRAM cells is also observable in the transient heating phase. The heating temporarily enforces the transistors to change the state. Note that this effect is only transient and the cells regenerate after cooling. The number of zero values increases during heating while the number of ones decreases accordingly. In our experiment, 820 bits changed their state to either 0 or 1. From these 820 bits, 257 bits were the same that also changed during

the long-term burn-in stress, i.e., we could identify 31.3 % of all bits that are apparently unstable.

How to Exploit NBTI Degradation? Many implementations of cryptographic algorithms store the secret key in non-volatile memory and load it into SRAM when needed. This key is then loaded always at the same memory location in SRAM because the program code or hardware implementation is usually given and not changeable. Hence, the key value gets "burned" into SRAM within a period of time, e.g., some weeks or months. The key can then be extracted by the following data-remanence attacks:

1. If an attacker is in possession of several implementations that store the same secret key, the attacker can recover parts of the internal SRAM of each implementation and can then average the obtained results to reveal all bits of the memory with high probability. This attack assumes that each implementation reveals the content of different SRAM cells.
2. If an attacker is in possession of only one implementation (or several implementations using different keys), he/she can first apply a burn-in stress test over several hours in order to artificially accelerate aging. As a second step, he/she can read out the preferred SRAM content. In order to identify useful bits, the attacker can mount a transient remanence attack to reveal *unstable* SRAM cells which potentially changed their state during the burn-in stress and which contain useful information about the secret key. All other cells are then considered as stable in this attack and provide no useful information. Finally, all recovered bits are used in partial-key recovery attacks.

5 Discussion and Further Research Suggestions

In the following, we discuss further research questions arisen by this work:

- The presented attacks have been performed on microcontrollers that were fabricated in rather old process technologies. Further research has to be done to evaluate the impact of thermal attacks on ICs with newer CMOS technologies.
- The temperature side channel provides a low-bandwidth characteristic. High-frequency leakages (e.g., containing data-dependent signals in the MHz or even GHz scale) can therefore not be directly exploited due to the low-pass filter characteristics. However, there exist implementations that use long operations like exponentiations in RSA that can create low-frequency signatures that are however exploitable (as demonstrated in [17,39] using acoustic signals in the kHz range).
- In order to validate our measurement setup, we also performed the measurements using a 0 Ohm resistor instead of a PT100 element. This is done because it naturally raises the question if the measured data really corresponds to the dissipated temperature or if it is caused by other side-channel sources, e.g., EM modulated signals. Using the resistor, however, we were

not able to identify any data-dependent signals so that we can exclude any signal interferences or the coupling of EM signals.
- We also characterized the temperature leakage of an 8-bit PIC16F84 microcontroller from Microchip Technology. We obtained similar results and could identify temperature-dependent processing of data. Details are given in Appendix A.2.
- Heat penetrates through different materials. Thus, the thermal conductivity of CMOS devices might be exploited in attacks where the heat conducts through EM shielding countermeasures (metal plates, mesh of power lines, sensors, etc).
- *How does the temperature affect power-analysis attacks?* This question has been answered by A. Vijaykumar [42] in her master's thesis. She evaluated temperature variation effects on Differential Power Analysis (DPA) attacks. By targeting KeeLoq and DES, she showed that the efficiency of DPA decreases with increased temperature due to decreasing power variances.
- By heating or cooling CMOS devices, the characteristics not only of memory but also of logic changes. Thermal attacks might even be used to circumvent countermeasure implementations, e.g., by increasing/decreasing the threshold voltages of watchdog implementations.
- The suitability of temperature attacks that indirectly exploit the leakage of *static* power consumption has to be investigated in future. Static power consumption is becoming more and more important as CMOS technology is shrinking. A static leak of an intermediate value of a cryptographic implementation creates a DC offset signal in the baseband that can be exploited in attacks that are provided with very low-bandwidth leakages from side channels such as the temperature or acoustic sound.

Acknowledgements. The work has been supported by the European Commission through the ICT program under contract ICT-SEC-2009-5-258754 (Tamper Resistant Sensor Node - TAMPRES), by the Austrian Science Fund (FWF) under the grant number TRP251-N23 (Realizing a Secure Internet of Things - ReSIT), and the European Cooperation in Science and Technology (COST) Action IC1204 (Trustworthy Manufacturing and Utilization of Secure Devices - TRUDEVICE).

References

1. D. Agrawal, B. Archambeault, J. R. Rao, and P. Rohatgi. The EM Side-channel(s). In B. S. Kaliski Jr., Ç. K. Koç, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers*, volume 2523 of *Lecture Notes in Computer Science*, pages 29–45. Springer, 2003.
2. J. Altet, A. Rubio, E. Schaub, S. Dilhaire, and W. Claeys. Thermal Coupling in Integrated Circuits: Application to Thermal Testing. *IEEE Journal of Solid-State Circuits*, 36(1):81–91, January 2001.

3. R. J. Anderson and M. G. Kuhn. Low Cost Attacks on Tamper Resistant Devices. In B. Christianson, B. Crispo, M. Lomas, and M. Roe, editors, *Security Protocols, 5th International Workshop*, volume 1361 of *Lecture Notes in Computer Science*, pages 125–136. Springer, 1997.
4. D. Asonov and R. Agrawal. Keyboard Acoustic Emanations. In *IEEE Symposium on Security and Privacym*, pages 3–11, 2004.
5. Atmel Corporation. ATmega 162/v Datasheet, 2003.
6. H. Bar-El, H. Choukri, D. Naccache, M. Tunstall, and C. Whelan. The Sorcerer's Apprentice Guide to Fault Attacks. Cryptology ePrint Archive (http://eprint.iacr.org/), Report 2004/100, 2004.
7. A. Barenghi, G. Bertoni, E. Parrinello, and G. Pelosi. Low Voltage Fault Attacks on the RSA Cryptosystem. In *Workshop on Fault Diagnosis and Tolerance in Cryptography – FDTC 2009, Lausanne, Switzerland, 2009. Proceedings.*, pages 23–31, 2009.
8. A. Barenghi, L. Breveglieri, I. Koren, and D. Naccache. Fault Injection Attacks on Cryptographic Devices: Theory, Practice and Countermeasures. *Proceedings of the IEEE*, 100(11):3056–3076, 2012.
9. D. Boneh, R. A. DeMillo, and R. J. Lipton. On the Importance of Checking Cryptographic Protocols for Faults (Extended Abstract). In W. Fumy, editor, *Advances in Cryptology - EUROCRYPT '97, International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, Germany, May 11-15, 1997, Proceedings*, volume 1233 of *Lecture Notes in Computer Science*, pages 37–51. Springer, 1997.
10. J. Brouchier, N. Dabbous, T. Kean, C. Marsh, and D. Naccache. Thermocommunication. eprint, 2009.
11. J. Brouchier, T. Kean, C. Marsh, and D. Naccache. Temperature Attacks. *Security Privacy, IEEE*, 7(2):79 –82, 2009.
12. C. Cakir, M. Bhargava, and K. Mai. 6T SRAM and 3T DRAM Data Retention and Remanence Characterization in 65nm bulk CMOS. In *Custom Integrated Circuits Conference – CICC 2012, USA, San Jose, 9-12 September, 2012*, pages 1–4, 2012.
13. D. Carluccio, K. Lemke, and C. Paar. Electromagnetic Side Channel Analysis of a Contactless Smart Card: First Results. In E. Oswald, editor, *Workshop on RFID and Lightweight Crypto (RFIDSec05), July 13-15, Graz, Austria*, pages 44–51, 2005.
14. M. Ershov, S. Saxena, H. Karbasi, S. Winters, S. Minehane, J. Babcock, R. Lindley, P. Clifton, M. Redford, and A. Shibkov. Dynamic Recovery of Negative Bias Temperature Instability in P-type MetalOxideSemiconductor Field-Effect Transistors. *Applied Physics Letters*, 83(8):1647–1649, 2003.
15. J. Ferrigno and M. Hlaváč. When AES Blinks: Introducing Optical Side Channel. *IET Information Security*, 2(3):94–98, June 2008.
16. K. Gandolfi, C. Mourtel, and F. Olivier. Electromagnetic Analysis: Concrete Results. In Ç. K. Koç, D. Naccache, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2001, Third International Workshop, Paris, France, May 14-16, 2001, Proceedings*, volume 2162 of *Lecture Notes in Computer Science*, pages 251–261. Springer, 2001.
17. D. Genkin, A. Shamir, and E. Tromer. RSA Key Extraction via Low-Bandwidth Acoustic Cryptanalysis. eprint, December 2013.
18. J. Giogetti, G. Scotti, A. Simonetti, and A. Trifiletti. Analysis of Data Dependance of Leakage Current in CMOS Cryptographic Hardware. In *Proceedings of the 17th ACM Great Lakes Symposium on VLSI, Stresa-Lago Maggiore, Italy, March 11-13, 2007*, pages 78–83. ACM, 2007.

19. S. Govindavajhala and A. W. Appel. Using Memory Errors to Attack a Virtual Machine. In *IEEE Symposium on Security and Privacy, Proceedings of the 2003*, pages 154–165, 2003.
20. P. Gutmann. Data Remanence in Semiconductor Devices. In *USENIX 2001 – Proceedings of the 10th Conference on USENIX Security Symposium, USA, Washington, D.C., August 1317, 2001*, Berkeley, CA, USA, 2001. USENIX Association.
21. J. Halderman, S. D.Schoen, N. Heninger, W. Clarkson, W. Paul, J. A.Calandrino, A. J.Feldman, J. Appelbaum, and E. W.Felten. Lest We Remember: Cold Boot Attacks on Encryption Keys. In *17th USENIX Security Symposium, San Jose, CA, July 2008*, pages 45–60, 2008.
22. M. Hutter, J.-M. Schmidt, and T. Plos. RFID and its Vulnerability to Faults. In E. Oswald and P. Rohatgi, editors, *Cryptographic Hardware and Embedded Systems – CHES 2008, 10th International Workshop, Washington DC, USA, August 10-13, 2008, Proceedings*, volume 5154 of *Lecture Notes in Computer Science*, pages 363–379. Springer, August 2008.
23. D. Karaklajíc, J.-M. Schmidt, and I. Verbauwhede. Hardware Designers Guide to Fault Attacks. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, pages 1–12, 2012.
24. P. C. Kocher. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In N. Koblitz, editor, *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, number 1109 in Lecture Notes in Computer Science, pages 104–113. Springer, 1996.
25. P. C. Kocher, J. Jaffe, and B. Jun. Differential Power Analysis. In *CRYPTO*, pages 388–397, 1999.
26. L. Lin and W. Burleson. Leakage-Based Differential Power Analysis (LDPA) on Sub-90nm CMOS Cryptosystems. In *ISCAS 2008 – IEEE International Symposium on Circuits and Systems, USA, Seattle, 18-21 May, 2008*, pages 252–255, 2008.
27. A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography*. Series on Discrete Mathematics and its Applications. CRC Press, 1997. ISBN 0-8493-8523-7, Available online at http://www.cacr.math.uwaterloo.ca/hac/.
28. A. Moradi. Side-Channel Leakage through Static Power - Should We Care about in Practice? eprint, January 2014.
29. T. Müller and M. Spreitzenbarth. FROST - Forensic Recovery of Scrambled Telephones. In M. Jacobson, M. Locasto, P. Mohassel, and R. Safavi-Naini, editors, *Applied Cryptography and Network Security–ACNS 2013, 11th International Conference, Banff, AB, Canada, June 25-28, 2013. Proceedings*, volume 7954, pages 373–388, 2011.
30. M. Otto. *Fault Attacks and Countermeasures*. PhD thesis, Universität Paderborn, 2005.
31. J.-J. Quisquater and D. Samyde. A new Tool for Non-Intrusive Analysis of Smart Cards Based on Electro-Magnetic Emissions, the SEMA and DEMA Methods,. Presented at the rump session of EUROCRYPT 2000, 2000.
32. J.-J. Quisquater and D. Samyde. Eddy Current for Magnetic Analysis with Active Sensor. In *Proceedings of the 3rd International Conference on Research in SmartCards (E-Smart'02), Nice, France, September, 2002*, pages 185–194. UCL, September 2002.
33. SageMath. Sage: Open source mathematics software system. http://sagemath.org, 2013.

34. D. Samyde, S. P. Skorobogatov, R. J. Anderson, and J.-J. Quisquater. On a New Way to Read Data from Memory. In *IEEE Security in Storage Workshop (SISW02)*, pages 65–69. IEEE Computer Society, 2002.
35. A. Schlösser, D. Nedospasov, J. Krämer, S. Orlic, and J.-P. Seifert. Simple Photonic Emission Analysis of AES. In E. Prouff and P. Schaumont, editors, *Cryptographic Hardware and Embedded Systems CHES 2012, 14th International Workshop, Leuven, Belgium, September 9-12, 2012. Proceedings*, volume 7428 of *LNCS*, pages 41–57, 2012.
36. J.-M. Schmidt and M. Hutter. Optical and EM Fault-Attacks on CRT-based RSA: Concrete Results. In K. C. Posch and J. Wolkerstorfer, editors, *Proceedings of Austrochip 2007, October 11, 2007, Graz, Austria*, pages 61–67. Verlag der Technischen Universität Graz, October 2007. ISBN 978-3-902465-87-0.
37. D. K. Schroder. Negative Bias Temperature Instability: What do we Understand? *Journal of Microelectronics Reliability*, 47(6):841–852, June 2006.
38. Sergei Skorobogatov. Using Optical Emission Analysis for Estimating Contribution to Power Consumption. In *Fault Diagnosis and Tolerance in Cryptography (FDTC)*, 2009.
39. A. Shamir and E. Tromer. Acoustic cryptanalysis - On nosy people and noisy machines. http://www.wisdom.weizmann.ac.il/~tromer/acoustic/. preliminary proof-of-concept presentation.
40. S. Skorobogatov. Low temperature data remanence in static RAM. Technical report, University of Cambridge Computer Laboratory, June 2002.
41. S. P. Skorobogatov. *Semi-invasive attacks - A new approach to hardware security analysis*. PhD thesis, University of Cambridge - Computer Laboratory, 2005. Available online at http://www.cl.cam.ac.uk/TechReports/.
42. A. Vijaykumar. DPA Resistance of Cryptographic Circuits Considering Temperature and Process Variations. Master's thesis, University of Cincinnati, Engineering and Applied Science: Computer Engineering, July 2012.
43. L. Zhuang, F. Zhou, and J. D. Tyga. Keyboard Acoustic Emanations Revisited. *ACM Transactions on Information and System Security*, 13(1):373–382, October 2009.

A Appendix

A.1 Attacking CRT-RSA using Faults

In the following, we consider an implementation of an RSA decryption that uses the Chinese Remainder Theorem (CRT) to speed up the computation. In our scenario, an adversary is able to supply the card with an input that is encrypted using textbook RSA and receives the decrypted message from the card. Further, the adversary is able to disturb the computation of this decryption and receives the result of this faulted computation. In order to describe how an adversary can benefit from this scenario to factor the modulus and thus compute the secret decryption key, we denote $n = pq$ an RSA modulus, where p and q are two large prime numbers. Let d be the private key and $e = d^{-1} \mod \varphi(n)$ the corresponding public exponent. Furthermore, $z = \text{CRT}\ (x, y)$ denotes the CRT recombination of the value $z \in \mathbf{Z}_n$ from values x, y of the subgroups $\mathbf{Z}_p$ and $\mathbf{Z}_q$ where

$$\text{CRT}\ (x, y) = xc_p + yc_q \mod n$$

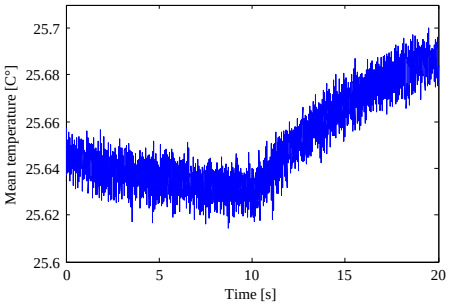

Fig. 10: Leakage of 0xFF in the second half of the acquisition window. No leakage during the first 10 seconds.

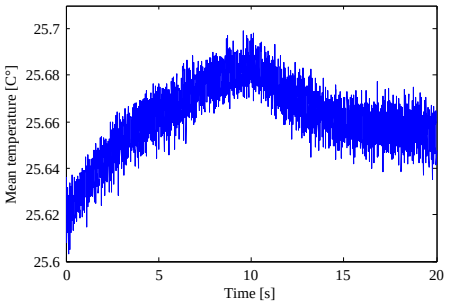

Fig. 11: Leakage of 0xFF in the first half of the acquisition window. The mean temperature decreases afterwards.

with $c_p = q\,(q^{-1} \mod p)$ and $c_q = p\,(p^{-1} \mod q)$ [27].

The usage of the CRT in this scenario allows computing two exponentiations in smaller sub-groups compared to a single exponentiations modulo n:

$$S \equiv \text{CRT}\,((m^d \mod p), (m^d \mod q)) \mod n.$$

The first fault attack that takes advantage of injecting a random fault Δ in this scenario was presented by Boneh et al. [9]. The fault Δ causes the device to output a value $\tilde{S}$ instead of S:

$$\begin{aligned}\tilde{S} &\equiv \text{CRT}\,((m \mod p)^d, (m \mod q)^d + \Delta) \mod n \\ &\equiv m^d + \Delta p\,(p^{-1} \mod q) \mod n.\end{aligned}$$

If an adversary gets hold of both a faulty $\tilde{S}$ and a correct signature S, the modulus n can be easily factorized by calculating $p = \gcd(\tilde{S} - S, n)$.

A.2 Temperature Leakage of a PIC16F84

We also investigated the leakage of a PIC16F84 microcontroller. We used the same measurement setup as described in Section 2 and measured the temperature on the decapsulated rear-side of the chip using a PT100 element. Instead of a `MOV` operation, we target an `ADD` instruction that adds either 0x00 or 0xFF to all internal registers that are previously initialized with zero. We measured 500 traces and averaged them to reduce noise.

Figure 10 shows the result where a zero value was written continuously over a period of 10 seconds. The value 0xFF is written afterwards for another 10 seconds. It shows an increase of temperature in the second half of the acquisition window. No leakage occurs in the first half of the trace. In Figure 11, the result is shown when 0xFF is written during the first 10 seconds, and zero is written afterwards. There, it shows that the temperature slowly increases, similarly to the second half of Figure 10. After 10 seconds, the temperature is decreasing again.