AN INFORMATION-THEORETIC MODEL FOR ADAPTIVE SIDE-CHANNEL ATTACKS

Boris Köpf, David Basin

ETH Zurich, Switzerland

SUCCESSFUL SIDE-CHANNEL ATTACKS

- Attacker must be able to extract information about the key through side-channel measurements.

SUCCESSFUL SIDE-CHANNEL ATTACKS

- Attacker must be able to extract information about the key through side-channel measurements.
- Attacker must be able to effectively recover the key from this information.

PROPOSAL

- A model to express the quantity of information that an adaptive attacker can extract from a system

PROPOSAL

- A model to express the quantity of information that an adaptive attacker can extract from a system
- Based on a definition of attack strategies, which are explicit representations of the adaptive decisions made by an attacker

PROPOSAL

- A model to express the quantity of information that an adaptive attacker can extract from a system
- Based on a definition of attack strategies, which are explicit representations of the adaptive decisions made by an attacker
- Expresses the attacker's expected uncertainty about the secret after they have performed a side-channel attack following a given strategy

ATTACK

- We have a function F which operates on a long-term constant secret key k

ATTACK

- We have a function F which operates on a long-term constant secret key k
- The malicious agent performs a series of attack steps in order to gather information for deducing k

ATTACK

- We have a function F which operates on a long-term constant secret key k
- The malicious agent performs a series of attack steps in order to gather information for deducing k
- The attack is adaptive if the observations used in the first n steps are used for calculating the $n+1$-th step

ATTACK STRATEGIES

MODEL FOR ATTACK STRATEGIES

- Use trees to define attack strategies, which capture the adaptive choices of the attacker

MODEL FOR ATTACK STRATEGIES

- Use trees to define attack strategies, which capture the adaptive choices of the attacker
- Two attack phases: query and response

ATTACK PHASES

Query phase

- Decide on message $m \in M$ with which to query the system

Response phase

- System responds with $f(k, m)$
- k isn't directly deducible

FORMAL MODEL FOR ATTACK STRATEGIES

A system under attack can be formalized as $f_I : K \times M \to O$, where K is the set of possible keys, M is the set of messages to which the system will respond and O is the set of observations the attacker can make. As the implementation I is constant, $f_I \sim f$.

Assuming full knowledge of the implementation of f and unbounded computing power...

- it is possible to deduce a set of keys that are **coherent** with the observation $f(k, m)$

FORMAL MODEL FOR ATTACK STRATEGIES

A system under attack can be formalized as $f_I : K \times M \rightarrow O$, where K is the set of possible keys, M is the set of messages to which the system will respond and O is the set of observations the attacker can make. As the implementation I is constant, $f_I \sim f$.

Assuming full knowledge of the implementation of f and unbounded computing power...

- it is possible to deduce a set of keys that are **coherent** with the observation $f(k, m)$
- A key $k \in K$ is **coherent** with $o \in O$ under $m \in M$ iff $f(k, m) = o$

FORMAL MODEL FOR ATTACK STRATEGIES

A system under attack can be formalized as $f_I : K \times M \to O$, where K is the set of possible keys, M is the set of messages to which the system will respond and O is the set of observations the attacker can make. As the implementation I is constant, $f_I \sim f$.

Assuming full knowledge of the implementation of f and unbounded computing power...

- it is possible to deduce a set of keys that are **coherent** with the observation $f(k, m)$
- A key $k \in K$ is **coherent** with $o \in O$ under $m \in M$ iff $f(k, m) = o$
- Two keys $k_1, k_2 \in K$ are **indistinguishable** under $m \in M$ iff $f(k_1, m) = f(k_2, m)$

FORMAL MODEL FOR ATTACK STRATEGIES

- **Key fact**: the set of keys coherent with the attacker's observation is the set that could have possibly led to this observation.

FORMAL MODEL FOR ATTACK STRATEGIES

- **Key fact**: the set of keys coherent with the attacker's observation is the set that could have possibly led to this observation.
- For every $m \in M$, **indistinguishability** under m is an equivalence relation on K.

- **Key fact**: the set of keys coherent with the attacker's observation is the set that could have possibly led to this observation.
- For every $m \in M$, **indistinguishability** under m is an equivalence relation on K.
- Every equivalence relation R on K corresponds to a partition P_R, and the equivalence classes of P_R are pairwise disjoint blocks B_i, such that $\bigcup_{i=1}^{r} B_i = K$

- **Key fact**: the set of keys coherent with the attacker's observation is the set that could have possibly led to this observation.
- For every $m \in M$, **indistinguishability** under m is an equivalence relation on K.
- Every equivalence relation R on K corresponds to a partition P_R, and the equivalence classes of P_R are pairwise disjoint blocks B_i, such that $\bigcup_{i=1}^{r} B_i = K$
- Taking f into account, $P_f = \{P_m \mid m \in M\}$, where P_m is induced by indistinguishability under m.

ATTACK PHASES, FORMALLY

Query phase

- Decide on message $m \in M$ with which to query the system

Response phase

- System responds with $f(k, m)$
- k isn't directly deducible

ATTACK PHASES, FORMALLY

Query phase

- Decide on a partition $P \in P_f$

Response phase

- The system reveals the block $B \in P$ that contains k

FORMALIZING ATTACK STRATEGIES, AN EXAMPLE

- Assuming a fixed set of partitions $\mathbb{P}$ on K, produce a tree of nodes labelled with subsets of K: the attacker's decisions with respect to the observations

FORMALIZING ATTACK STRATEGIES, AN EXAMPLE

- Assuming a fixed set of partitions $\mathbb{P}$ on K, produce a tree of nodes labelled with subsets of K: the attacker's decisions with respect to the observations
- $K = \{1, 2, 3, 4\}$

FORMALIZING ATTACK STRATEGIES, AN EXAMPLE

- Assuming a fixed set of partitions $\mathbb{P}$ on K, produce a tree of nodes labelled with subsets of K: the attacker's decisions with respect to the observations
- $K = \{1, 2, 3, 4\}$
- $P_R = \{\{\{1\}, \{2, 3, 4\}\}, \{\{1, 2\}, \{3, 4\}\}, \{\{\{1, 2, 3\}, \{4\}\}\}$

- Assuming a fixed set of partitions $\mathbb{P}$ on K, produce a tree of nodes labelled with subsets of K: the attacker's decisions with respect to the observations
- $K = \{1, 2, 3, 4\}$
- $P_R = \{\{\{1\}, \{2, 3, 4\}\}, \{\{1, 2\}, \{3, 4\}\}, \{\{\{1, 2, 3\}, \{4\}\}\}$
- Suppose the attacker picks $\{\{1, 2\}, \{3, 4\}\}$ as their first query

FORMALIZING ATTACK STRATEGIES, AN EXAMPLE

- Assuming a fixed set of partitions $\mathbb{P}$ on K, produce a tree of nodes labelled with subsets of K: the attacker's decisions with respect to the observations
- $K = \{1, 2, 3, 4\}$
- $P_R = \{\{\{1\}, \{2, 3, 4\}\}, \{\{1, 2\}, \{3, 4\}\}, \{\{\{1, 2, 3\}, \{4\}\}\}$
- Suppose the attacker picks $\{\{1, 2\}, \{3, 4\}\}$ as their first query
- If the system responds with $\{1, 2\}$, the attacker chooses $\{\{1\}, \{2, 3, 4\}\}$ as their next query

FORMALIZING ATTACK STRATEGIES, AN EXAMPLE

- Assuming a fixed set of partitions $\mathbb{P}$ on K, produce a tree of nodes labelled with subsets of K: the attacker's decisions with respect to the observations
- $K = \{1, 2, 3, 4\}$
- $P_R = \{\{\{1\}, \{2, 3, 4\}\}, \{\{1, 2\}, \{3, 4\}\}, \{\{\{1, 2, 3\}, \{4\}\}\}$
- Suppose the attacker picks $\{\{1, 2\}, \{3, 4\}\}$ as their first query
- If the system responds with $\{1, 2\}$, the attacker chooses $\{\{1\}, \{2, 3, 4\}\}$ as their next query
- Otherwise, they choose $\{\{1, 2, 3\}, \{4\}\}$.

FORMALIZING ATTACK STRATEGIES, AN EXAMPLE

- Assuming a fixed set of partitions $\mathbb{P}$ on K, produce a tree of nodes labelled with subsets of K: the attacker's decisions with respect to the observations
- $K = \{1, 2, 3, 4\}$
- $P_R = \{\{\{1\}, \{2, 3, 4\}\}, \{\{1, 2\}, \{3, 4\}\}, \{\{\{1, 2, 3\}, \{4\}\}\}$
- Suppose the attacker picks $\{\{1, 2\}, \{3, 4\}\}$ as their first query
- If the system responds with $\{1, 2\}$, the attacker chooses $\{\{1\}, \{2, 3, 4\}\}$ as their next query
- Otherwise, they choose $\{\{1, 2, 3\}, \{4\}\}$.
- This way, they can determine any key in two steps.

QUANTITATIVE EVALUATION OF ATTACK STRATEGIES

INTUITION

- The attacker gains more information (reduces the uncertainty) about the key as the attack strategy is refined

INTUITION

- The attacker gains more information (reduces the uncertainty) about the key as the attack strategy is refined
- Different measures of entropy correspond to different notions of resistance against brute-force guessing of the key, therefore, also attack strategy generation

INTUITION

- The attacker gains more information (reduces the uncertainty) about the key as the attack strategy is refined
- Different measures of entropy correspond to different notions of resistance against brute-force guessing of the key, therefore, also attack strategy generation
- The paper presents Shannon entropy H, guessing entropy G, and marginal guesswork W_α and builds a model of quantitative evaluation on top of them

ENTROPY 101

- The attacker assumes a probability measure p on K

ENTROPY 101

- The attacker assumes a probability measure p on K
- The Shannon entropy of a random variable $X : K \to \mathcal{X}$, given $p \sim p_X$, is defined as:

$$H(X) = -\sum_{x \in \mathcal{X}} p(x) log_2 p(x)$$

ENTROPY 101

- The attacker assumes a probability measure p on K
- The Shannon entropy of a random variable $X : K \to \mathcal{X}$, given $p \sim p_X$, is defined as:

$$H(X) = -\sum_{x \in \mathcal{X}} p(x) log_2 p(x)$$

- Formally, it is a lower bound for the average number of bits required for representing the results of independent repetitions of the experiment associated with X

ENTROPY 101

- The attacker assumes a probability measure p on K
- The Shannon entropy of a random variable $X : K \to \mathcal{X}$, given $p \sim p_X$, is defined as:

$$H(X) = -\sum_{x \in \mathcal{X}} p(x) log_2 p(x)$$

- Formally, it is a lower bound for the average number of bits required for representing the results of independent repetitions of the experiment associated with X
- Informally, it measures how surprised you expect to be on average after sampling the random variable X

- Given another random variable, $Y : K \to \mathcal{Y}$, **conditional entropy** $H(X \mid Y)$ is defined as the expected value of $H(X \mid Y = y)$, for all $y \in \mathcal{Y}$.

- Given another random variable, $Y : K \to \mathcal{Y}$, **conditional entropy** $H(X \mid Y)$ is defined as the expected value of $H(X \mid Y = y)$, for all $y \in \mathcal{Y}$.
- Formally,

$$H(X \mid Y) = \sum_{y \in \mathcal{Y}} p_Y(y) H(X \mid Y = y)$$

SHANNON ENTROPY: HOW DO WE USE THIS?

- Consider now an attack strategy **a** and the corresponding variables U (random choice of a key in K) and V_a (random choice of the enclosing block $B \in P_a$).

SHANNON ENTROPY: HOW DO WE USE THIS?

- Consider now an attack strategy a and the corresponding variables U (random choice of a key in K) and V_a (random choice of the enclosing block $B \in P_a$).
- The initial uncertainty about the key can be written as $H(U)$

SHANNON ENTROPY: HOW DO WE USE THIS?

- Consider now an attack strategy **a** and the corresponding variables U (random choice of a key in K) and V_a (random choice of the enclosing block $B \in P_a$).
- The initial uncertainty about the key can be written as $H(U)$
- After getting the system's response, that is, the block $B \in P_a$ which encloses the key, the remaining uncertainty is:

$$H(U \mid V_a = B)$$

SHANNON ENTROPY: HOW DO WE USE THIS?

- Consider now an attack strategy **a** and the corresponding variables U (random choice of a key in K) and V_a (random choice of the enclosing block $B \in P_a$).
- The initial uncertainty about the key can be written as $H(U)$
- After getting the system's response, that is, the block $B \in P_a$ which encloses the key, the remaining uncertainty is:

$$H(U \mid V_a = B)$$

- Neat!

GUESSING ENTROPY

- The guessing entropy of a random variable X is the average number of questions of the kind ”does $X = x$ hold?” that must be asked to guess the value of X correctly

GUESSING ENTROPY

- The guessing entropy of a random variable X is the average number of questions of the kind ”does $X = x$ hold?” that must be asked to guess the value of X correctly
- The **guessing entropy** is:

$$G(X) = \sum_{1 \leq i \leq |\mathcal{X}|} i \, p(x_i)$$

GUESSING ENTROPY

- The guessing entropy of a random variable X is the average number of questions of the kind "does $X = x$ hold?" that must be asked to guess the value of X correctly
- The **guessing entropy** is:

$$G(X) = \sum_{1 \leq i \leq |\mathcal{X}|} i\, p(x_i)$$

- The conditional guessing entropy is the expected number of optimal guesses needed to determine X when Y is known

- The guessing entropy of a random variable X is the average number of questions of the kind "does $X = x$ hold?" that must be asked to guess the value of X correctly
- The **guessing entropy** is:

$$G(X) = \sum_{1 \leq i \leq |\mathcal{X}|} i\, p(x_i)$$

- The conditional guessing entropy is the expected number of optimal guesses needed to determine X when Y is known
- The **conditional guessing entropy** is:

$$G(X \mid Y) = \sum_{y \in \mathcal{Y}} p_Y(y)\, G(X \mid Y = y)$$

GUESSING ENTROPY: HOW DO WE USE THIS?

The guessing entropy $G(U \mid V_a)$ is a lower bound on the expected number of off-line guesses that an attacker must perform for key recovery after carrying out a side-channel attack using the strategy **a** .

MARGINAL GUESSWORK

- For some fixed $\alpha \in [0, 1]$, the marginal guesswork of a random variable X quantifies the number of questions of the kind ”does $X = x$ hold?” that must be asked to determine X with an α chance of success.

- For some fixed $\alpha \in [0, 1]$, the marginal guesswork of a random variable X quantifies the number of questions of the kind ”does $X = x$ hold?” that must be asked to determine X with an α chance of success.
- Formally, the α-**marginal guesswork of** X is:

$$W_\alpha(X) = min\{j \mid \sum_{1 \leq i \leq j} p(x_i) \geq \alpha\}$$

MARGINAL GUESSWORK

- For some fixed $\alpha \in [0, 1]$, the marginal guesswork of a random variable X quantifies the number of questions of the kind "does $X = x$ hold?" that must be asked to determine X with an α chance of success.
- Formally, the α-**marginal guesswork of** X is:

$$W_\alpha(X) = min\{j \mid \sum_{1 \leq i \leq j} p(x_i) \geq \alpha\}$$

- Completely analogously to the guessing entropy, we can define the $W_\alpha(X|Y)$ as the **conditional** α**-marginal guesswork**

MARGINAL GUESSWORK: HOW DO WE USE THIS?

The conditional α-marginal guesswork, $W_\alpha(U \mid V_a)$ is a lower-bound on the expected number of guesses that an attacker needs to perform in order to determine the key with an α chance of success, after having carried out a side-channel attack using strategy **a** .

WORST-CASE ENTROPY MEASURES

- These average case measurements can be extended into worst-case measurements by quantifying the guessing effort for the keys in K that are easiest to guess

- These average case measurements can be extended into worst-case measurements by quantifying the guessing effort for the keys in K that are easiest to guess
- The **minimal guessing entropy** is defined as:

$$\hat{G}(U \mid V_a) = min\{G(U \mid V_a = B) \mid B \in P_a\}$$

WORST-CASE ENTROPY MEASURES

- These average case measurements can be extended into worst-case measurements by quantifying the guessing effort for the keys in K that are easiest to guess
- The **minimal guessing entropy** is defined as:

$$\hat{G}(U \mid V_a) = min\{G(U \mid V_a = B) \mid B \in P_a\}$$

- Average case measurements are better suited for distinguishing between partiitons

MEASURING THE RESISTANCE TO OPTIMAL ATTACKS

RESISTANCE TO ATTACKS

- There is a trade-off between the number of attack steps taken and the attacker's uncertainty about the key

RESISTANCE TO ATTACKS

- There is a trade-off between the number of attack steps taken and the attacker's uncertainty about the key
- Using the different entropy measurements, let's define $\Phi_{\mathcal{E}}(n)$, parametrized by $\mathcal{E} \in \{H, G, W_\alpha\}$, whose value is the expected remaining uncertainty after n steps of an optimal attack strategy.

RESISTANCE TO ATTACKS

- There is a trade-off between the number of attack steps taken and the attacker's uncertainty about the key
- Using the different entropy measurements, let's define $\Phi_{\mathcal{E}}(n)$, parametrized by $\mathcal{E} \in \{H, G, W_\alpha\}$, whose value is the expected remaining uncertainty after n steps of an optimal attack strategy.
- $\Phi_{\mathcal{E}}(n)$ can be used for assessing the implementation's vulnerability to side-channel attacks

- Worst case: the attacker is proceeding optimally

ASSESSING VULNERABILITY

- Worst case: the attacker is proceeding optimally
- A strategy **a** is optimal with respect to $\mathcal{E} \in \{H, G, W_\alpha\}$ iff $\mathcal{E}(U \mid V_a) \leq \mathcal{E}(U | V_b)$, for all strategies **b** , of the same length as **a**

ASSESSING VULNERABILITY

- Worst case: the attacker is proceeding optimally
- A strategy **a** is optimal with respect to $\mathcal{E} \in \{H, G, W_\alpha\}$ iff $\mathcal{E}(U \mid V_a) \leq \mathcal{E}(U | V_b)$, for all strategies **b** , of the same length as **a**
- The **resistance to an optimal attack**, $\Phi_\mathcal{E}(n)$ is then:

$$\Phi_\mathcal{E}(n) = \mathcal{E}(U \mid V_o)$$

where **o** is the optimal attack of length n with respect to $\mathcal{E}$.

ASSESSING VULNERABILITY

- Worst case: the attacker is proceeding optimally
- A strategy **a** is optimal with respect to $\mathcal{E} \in \{H, G, W_\alpha\}$ iff $\mathcal{E}(U \mid V_a) \leq \mathcal{E}(U|V_b)$, for all strategies **b** , of the same length as **a**
- The **resistance to an optimal attack**, $\Phi_{\mathcal{E}}(n)$ is then:

$$\Phi_{\mathcal{E}}(n) = \mathcal{E}(U \mid V_o)$$

  where **o** is the optimal attack of length n with respect to $\mathcal{E}$.
- The paper formally justifies the intuition that more attack steps lead to less uncertainty about the key by proving that $\Phi_{\mathcal{E}}$ decreases monotonously with n.

AUTOMATED VULNERABILITY ANALYSIS

COMPUTABILITY OF RESISTANCE TO ATTACKS, ASSUMPTIONS

- Let $\mathbb{P}$ be a set of partitions over K and $r \geq 2$ be the maximum number of blocks of a partition

COMPUTABILITY OF RESISTANCE TO ATTACKS, ASSUMPTIONS

- Let $\mathbb{P}$ be a set of partitions over K and $r \geq 2$ be the maximum number of blocks of a partition
- The sets O and K are ordered, and comparing elements within them costs $\mathcal{O}(1)$

- Given $f : K \times M \to O$, one can build the partitions (as disjoint-set data structures) for $\mathbb{P}_f$ in

$$\mathcal{O}(|M||K|\ log|K|)$$

time, assuming that f can be computed in $\mathcal{O}(1)$

COMPUTABILITY OF RESISTANCE TO ATTACKS

- Given $f : K \times M \to O$, one can build the partitions (as disjoint-set data structures) for $\mathbb{P}_f$ in

$$\mathcal{O}(|M||K|\ log|K|)$$

  time, assuming that f can be computed in $\mathcal{O}(1)$
- Using brute-force optimal attack searching, $\Phi_{\mathcal{E}}(n)$ can be computed in

$$\mathcal{O}(n|M|^{r^n}|K|\ log|K|)$$

  under the assumption that $\mathcal{E}$ can be computed in $\mathcal{O}(|K|)$

- Given $f : K \times M \to O$, one can build the partitions (as disjoint-set data structures) for $\mathbb{P}_f$ in

$$\mathcal{O}(|M||K|\ log|K|)$$

time, assuming that f can be computed in $\mathcal{O}(1)$

- Using brute-force optimal attack searching, $\Phi_{\mathcal{E}}(n)$ can be computed in

$$\mathcal{O}(n|M|^{r^n}|K|\ log|K|)$$

under the assumption that $\mathcal{E}$ can be computed in $\mathcal{O}(|K|)$

- This is useless.

GREEDY HEURISTIC

- Consider an attacker who has performed a number of attack steps against a set of partitions $\mathbb{P}$ and has narrowed down the set of possible keys to a subset $A \subseteq K$.

GREEDY HEURISTIC

- Consider an attacker who has performed a number of attack steps against a set of partitions $\mathbb{P}$ and has narrowed down the set of possible keys to a subset $A \subseteq K$.
- A greedy choice for the next query is a partition $P \in \mathbb{P}$ that minimizes the remaining entropy, similar to what was done to get the minimal guessing entropy.

GREEDY HEURISTIC

- Consider an attacker who has performed a number of attack steps against a set of partitions $\mathbb{P}$ and has narrowed down the set of possible keys to a subset $A \subseteq K$.
- A greedy choice for the next query is a partition $P \in \mathbb{P}$ that minimizes the remaining entropy, similar to what was done to get the minimal guessing entropy.
- The greedy $\hat{\Phi}_{\mathcal{E}}(n)$ is an approximation, thus it will not have the same entropy as $\Phi_{\mathcal{E}}(n)$, but will, in general, converge.

GREEDY HEURISTIC

- Consider an attacker who has performed a number of attack steps against a set of partitions $\mathbb{P}$ and has narrowed down the set of possible keys to a subset $A \subseteq K$.
- A greedy choice for the next query is a partition $P \in \mathbb{P}$ that minimizes the remaining entropy, similar to what was done to get the minimal guessing entropy.
- The greedy $\hat{\Phi}_{\mathcal{E}}(n)$ is an approximation, thus it will not have the same entropy as $\Phi_{\mathcal{E}}(n)$, but will, in general, converge.
- The value $\hat{\Phi}_{\mathcal{E}}(n)$ can be computed in

$$\mathcal{O}(n\, r\, |M|\, |K|^2)$$

under the assumption that $\mathcal{E}$ can be computed in $\mathcal{O}(|K|)$.

GREEDY HEURISTIC IMPLEMENTATION

```
greedy :: [Part k] -> Int -> [k] -> Part k
greedy f n keys = app n (greedystep f) [keys]

greedystep :: [Part k] -> Part k -> Part k
greedystep f pt = concat (map refine pt)
  where refine b = minimumBy order (restrict b f)
```

EXPERIMENTS

TIMING BEHAVIOUR OF ARITHMETIC CIRCUITS

- Circuits for multiplying integers

TIMING BEHAVIOUR OF ARITHMETIC CIRCUITS

- Circuits for multiplying integers
- Circuits for multiplication and exponentiation in finite fields $\mathbb{F}_{2^w}$

TIMING BEHAVIOUR OF ARITHMETIC CIRCUITS

- Circuits for multiplying integers
- Circuits for multiplication and exponentiation in finite fields $\mathbb{F}_{2^w}$
- Useful in many encryption schemes, decryption usually consists of exponentiation followed by multiplication

A FEASIBLE APPROXIMATION

- The chosen entropy is the guessing entropy $\mathcal{E} = G$, and the approximation $\hat{\Phi}_{\mathcal{E}}$ is used, with a further parameterization:

A FEASIBLE APPROXIMATION

- The chosen entropy is the guessing entropy $\mathcal{E} = G$, and the approximation $\hat{\Phi}_{\mathcal{E}}$ is used, with a further parameterization:
- the bit-width w of the operands of each algorithm is proposed, as bit-regularity in the values of Φ for $w \in \{2, \ldots, w_{max}\}$ is assumed to show structural similarities of the algorithms

A FEASIBLE APPROXIMATION

- The chosen entropy is the guessing entropy $\mathcal{E} = G$, and the approximation $\hat{\Phi}_{\mathcal{E}}$ is used, with a further parameterization:
- the bit-width w of the operands of each algorithm is proposed, as bit-regularity in the values of Φ for $w \in \{2, \ldots, w_{max}\}$ is assumed to show structural similarities of the algorithms
- The approximation $\hat{\Phi}^{w}_{\mathcal{E}}$ can now be extrapolated for $w \geq w_{max}$ which would have been infeasible otherwise.

SETUP

- For each bit-width $w \in \{2, \ldots, 8\}$, the circuit simulator built value tables for the side-channel $f\colon \{0,1\}^w \times \{0,1\}^w \to O$, with $O \equiv \mathbb{N}$ representing the observation of the number of clock-ticks until termination

SETUP

- For each bit-width $w \in \{2, \ldots, 8\}$, the circuit simulator built value tables for the side-channel $f\colon \{0,1\}^w \times \{0,1\}^w \to O$, with $O \equiv \mathbb{N}$ representing the observation of the number of clock-ticks until termination
- The doubling and addition operations used in the implementation of integer multiplication each take one clock cycle

SETUP

- For each bit-width $w \in \{2, \ldots, 8\}$, the circuit simulator built value tables for the side-channel $f\colon \{0,1\}^w \times \{0,1\}^w \to O$, with $O \equiv \mathbb{N}$ representing the observation of the number of clock-ticks until termination
- The doubling and addition operations used in the implementation of integer multiplication each take one clock cycle
- As the algorithm does work only if a bit is 1, the running time reflects the number of 1-bits in k, that is, k's Hamming weight.

SETUP

- For each bit-width $w \in \{2, \ldots, 8\}$, the circuit simulator built value tables for the side-channel $f\colon \{0,1\}^w \times \{0,1\}^w \to O$, with $O \equiv \mathbb{N}$ representing the observation of the number of clock-ticks until termination
- The doubling and addition operations used in the implementation of integer multiplication each take one clock cycle
- As the algorithm does work only if a bit is 1, the running time reflects the number of 1-bits in k, that is, k's Hamming weight.
- The Hamming weight defines the equivalence relation over K

SETUP

- For each bit-width $w \in \{2, \ldots, 8\}$, the circuit simulator built value tables for the side-channel $f\colon \{0,1\}^w \times \{0,1\}^w \to O$, with $O \equiv \mathbb{N}$ representing the observation of the number of clock-ticks until termination
- The doubling and addition operations used in the implementation of integer multiplication each take one clock cycle
- As the algorithm does work only if a bit is 1, the running time reflects the number of 1-bits in k, that is, k's Hamming weight.
- The Hamming weight defines the equivalence relation over K
- Operations in $\mathbb{F}_{2^w}$ more complicated due to nested loops

EXPERIMENT CONCLUSION

One timing measurement reveals a quantity of information larger than that contained in the Hamming weight, but it does not completely determine the key. A second measurement, however, can reveal all remaining key information.

CONCLUSION

- The solution depends on enumerating the keyspace, thus does not scale

CONCLUSION

- The solution depends on enumerating the keyspace, thus does not scale
- Extrapolation is possible, but the approximation is very limited, and generally doesn't work

CONCLUSION

- The solution depends on enumerating the keyspace, thus does not scale
- Extrapolation is possible, but the approximation is very limited, and generally doesn't work
- Structural regularity is useful for parameterized algorithms

CONCLUSION

- The solution depends on enumerating the keyspace, thus does not scale
- Extrapolation is possible, but the approximation is very limited, and generally doesn't work
- Structural regularity is useful for parameterized algorithms
- Noise can be taken care of by increasing the number of measurements, or introducing noise models

QUESTIONS?