A Coding-Theoretic Approach to Recovering Noisy RSA Keys

K.G. Paterson, A. Polychroniadou & D.L. Sibborn

Royal Holloway, University of London

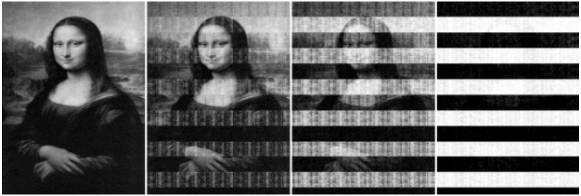
4 December, 2012

Cold Boot

- Usenix 2008 - Halderman et al. noted that DRAMs retain their contents for a while after power is lost.
- Bits in memory can be extracted, but they will have errors.
- 0 bits will always flip with very low probability (<1%), but 1 bits will flip with much higher probability which increases with time.
- For example

Original memory:	11000101101101001 . . .
Noisy memory:	11100001100100001 . . .

Cold Boot Attacks



- Why is this a problem?
- Secrets may be stored in memory.

The Big Question

Given a noisy RSA key obtained from a cold boot attack, how can we recover the original key?

Heninger & Shacham (HS) Algorithm (Crypto 2009)

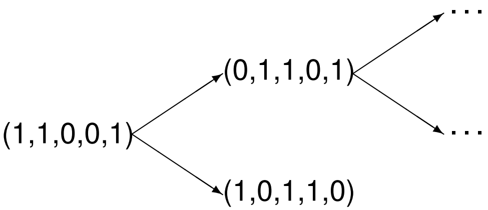
- A PKCS #1 RSA key has the form $(N, e, p, q, d, d_p, d_q, q_p^{-1})$.
- The HS algorithm assumes a noisy PKCS # 1 RSA key has been obtained and some bits of the RSA key are known to be correct.
- The HS algorithm uses algebraic relations between the bits of $sk = (p, q, d, d_p, d_q)$ to generate possible solutions for the next set of bits of the original key.

$$\begin{aligned}
p[i] + q[i] &= c_1 \mod 2\\
d[i + \tau(k)] + p[i] + q[i] &= c_2 \mod 2\\
d_p[i + \tau(k_p)] + p[i] &= c_3 \mod 2\\
d_q[i + \tau(k_q)] + q[i] &= c_4 \mod 2,
\end{aligned}$$

HS Expansion Phase

Starting bits

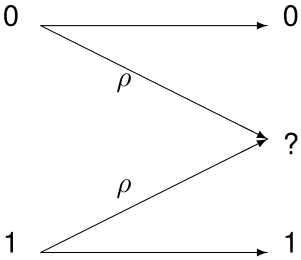
Known bits:
(0,?,1,?,?)



- Given enough time the algorithm always recovers the original key.

HS Algorithm

- From our perspective, the HS algorithm considers a key degraded according to an erasure channel:

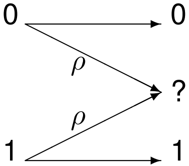


- If $\rho < 0.73$ the algorithm is provably efficient with high probability.

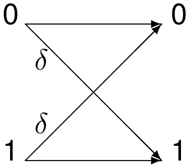
Henecka, May & Meurer (HMM) Algorithm (Crypto 2010)

- HMM assume that each bit of *sk* can flip with probability δ.
- The HMM algorithm considers 2^t sets of candidate solutions on $5t$ bits obtained by solving the HS equations on t consecutive positions.
- For each candidate solution on $5t$ bits the HMM algorithm counts the number of bit matches with the noisy RSA key and discards a candidate if there are less than C matches.
- The expansion and pruning phases are iterated on remaining candidates until we have recovered solutions across all bits of the RSA key.
- Asymptotically, when $\delta < 0.237$ the algorithm is provably efficient and recovers *sk* with reasonable success probability.

The Three (Implicit) Models

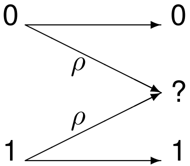


HS: $\rho < 0.73$.

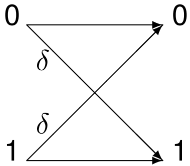


HMM: $\delta < 0.237$.

The Three (Implicit) Models

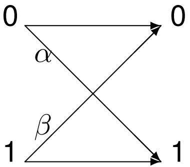


HS: $\rho < 0.73$.



HMM: $\delta < 0.237$.

These channels are not appropriate for cold boot!



Cold Boot: $\alpha \ll \beta$.

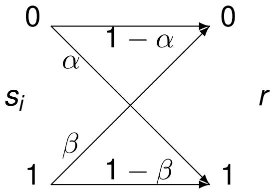
Questions We Address

Questions

- HS is provably efficient when $\rho < 0.73$ and HMM is provably efficient when bits flip with probability $\delta < 0.237$. Is there an underlying explanation for these constants?
- Can the results be improved further, and are there ultimate noise limits which no algorithm can handle?
- Can we design an algorithm that is applicable to the motivating cold boot scenario?

Our Perspective

- We view the situation as a problem in coding theory.
- We consider the set $\{s_i\}_{i \in I}$ of partial candidates as a code. One of these s_i is the correct RSA key which is degraded when retrieved via a cold boot attack.

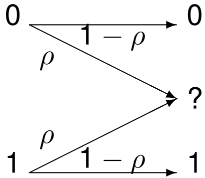


Our Perspective

- If we have obtained r via a cold boot attack, we wish to decode and identify the key s_j that was degraded.
- We are able to use standard results such as Shannon's noisy channel coding theorem to derive bounds on efficiency.
- This perspective enables us to analyse realistic cold boot attacks.

Erasure Channel

- The HS algorithm is concerned with the erasure channel.



- The capacity of this channel is $1 - \rho$.
- The rate of the code is 0.2.

Erasure Channel

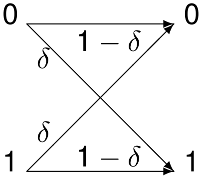
- The converse to Shannon's noisy channel coding theorem says that no algorithm that outputs a single codeword can reliably decode *r* when $1 - \rho \leq 0.2$.
- Hence, for reliable decoding we must have $\rho < 0.8$.
- By contrast, HS managed $\rho < 0.73$.

Key Result

For list decoding it can be shown that, on average, an exponential list of candidates will need to be considered when the code rate exceeds capacity.

Binary Symmetric Channel

- The algorithm of HMM is a decoding procedure for the binary symmetric channel.



- The capacity is $C = 1 - H_2(\delta)$.
- The code rate is at least 0.2.

Binary Symmetric Channel

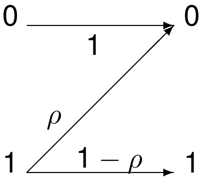
- Applying Shannon's theorem, an algorithm that outputs a single codeword cannot reliably decode when $1 - H_2(\delta) \leq 0.2$.
- Hence, only $\delta < 0.243$ is feasible.
- Note that HMM can handle $\delta < 0.237$.

Key Result

When $\delta \geq 0.243$ it can be shown that no algorithm can list decode using a polynomially-sized list.

Z-channel

- An idealised version of a cold boot attack can be modelled by a Z-channel.



- The HS analysis can be applied to this channel to show that an algorithm keeping **all** 'correct solutions' will be efficient when $\rho < 0.46$.
- The capacity bound on ρ for this channel is approximately 0.666.

Our New Algorithm

- From all the candidate solutions $\{s_i\}_{i \in I}$ we wish to find

$$\max_i \mathbb{P}(\mathrm{s}_i \mid \mathrm{r}),$$

where r is the noisy RSA key.

- Using Bayes' theorem and the assumption that $\mathbb{P}(\mathrm{s}_i)$ is equal for all i, this is equivalent to finding

$$\max_i \mathbb{P}(\mathrm{r} \mid \mathrm{s}_i).$$

- This can be calculated as

$$\max_i \left((1-\alpha)^{n^i_{00}} \alpha^{n^i_{01}} (1-\beta)^{n^i_{11}} \beta^{n^i_{10}} \right).$$

- We keep the L candidates with the greatest likelihood.

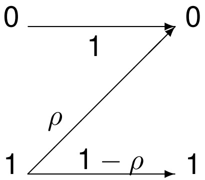
Experiments

- We will shortly see our experimental results for the Z-channel, the cold boot channel and the binary symmetric channel.
- For each experiment we degraded 100 RSA keys (where each modulus length is 1024 bits) according to the relevant channel.
- We then used our maximum-likelihood algorithm to attempt to recover the noisy RSA keys.

Idealised Cold Boot

- The capacity bound for ρ is 0.666.

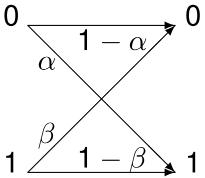
ρ	0.3	0.4	0.46	0.5	0.55	0.6	0.63
Success	1	0.98	0.87	0.81	0.43	0.13	0.03



Cold Boot Scenario

- For these experiments we set $\alpha = 0.001$. The capacity bound for this channel is $\beta = 0.658$.

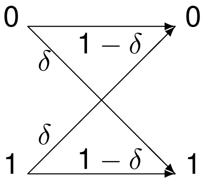
β	0.2	0.3	0.4	0.5	0.55	0.6	0.61
Success	1	0.97	0.97	0.66	0.31	0.09	0.04



Henecka, May & Meurer Setting

- The capacity bound for δ is 0.243.

δ	0.08	0.12	0.16	0.2	0.21	0.22
HMM	0.5	0.5	0.35	0.21	-	-
Us	1	0.93	0.84	0.20	0.08	0.04



Conclusions

- We have considered a more general setting than HS and HMM, which allows us to model true cold boot attacks.
- We have presented a new algorithm that, for practical RSA key sizes, outperforms the HS and HMM algorithms and is applicable to the true cold boot setting.
- We have explored the connections between the cold boot problem and coding theory, using the connections to give bounds on performance and to inspire our new algorithm.