6.857: Computer and Network Security (Spring 2009)

Guest lecturer: Eran Tromer

Lecture 7: Generic Attacks and Large-Scale Cryptanalysis

February 25, 2009

1 Time/memory tradeoff for function inversion

- Function Inversion Problem: let $f : \{0, \ldots, N-1\} \to \{0,1\}^n$. Given $y \in \{0, \ldots, N-1\}$, find a preimage $x \in \{0, \ldots, N-1\}$ such that $f(x) = y$.
- Cryptanalytic applications:
  - Break MACs and signatures: find a hash function preimage via $f(x) = \text{SHA1}(x)$
  - Break encryption: given the AES encryption of m, find the key: $f(x) = \text{AES}_x(m)$
  - Recover passwords from their hashes (Unix /etc/passwd file, Word passwords, SMB file share passwords)
  - Most general: "one-way functions" which underlie all of cryptography
- Trivial algorithms
  - Exhaustive search: time $T \approx N$, memory $M \approx 1$
  - Exhaustive table:
    * Off-line preprocessing (just once!)
    * Memory $M \approx N$
    * On-line time: $T \approx 1$
  - For $N = 2^{64}$: 2^{64} nanoseconds = 584 years / 2^{64} bytes = 16 exibytes
- Hellman's Time/Memory Tradeoff [Hellman 1980]
  - For f that is a single-cycle permutation
    * Off-line: pick t and x_0, compute a table $(f^{it}(x_0))_{i=0,\ldots N/t-1}$. Memory: $M = N/t$
    * On-line: compute $f^j(y)$ for increasing j until you hit f^{it} in the table, then output $f^{(i-1)t+j-1}$. Time: $M = t$. Tradeoff: $TM = N$.
  - For random f, naive version:

- Off-line: pick m random start points $x_0, \ldots, x_{m-1}$and chain length t. Traverse each chain and save a table $(m_i, f^t(m_i))$ indexed by end point. Memory: $M \approx mt$.
  - On-line: traverse from y until the end of the chain (table hit), then traverse that chain from the beginning.
  - Problem: table must "cover" most of $\{0, \ldots, N-1\}$ but it's difficult to cover more than N/t values:
    Once we have t rows covering $mt > N/t$ values, a new row of t elements is likely to collide (Birthday paradox: $mt \cdot t > N$).

- For random f, naive version:
  - Build t different tables using t functions $f_0, \ldots, f_{t-1}$, such that each f_k induces a different graph structure, but inverting f_k suffices for inverting f.
    - Example: $f_k(x) = f(x \oplus k)$. (This is heuristic, and in this case will fail is f ignores the $\log_2 t$ least-significant bits of its input).
  - Empirically: with $mt^2 \approx N$, each table covers about $0.8mt$ values, and t tables cover about $0.55N$.
  - Memory: $M \approx mt$. Time: $T \approx t^2$. Hence $TM^2 \approx m^2t^4 \approx N^2$
  - Tradeoff: $\underline{TM^2 = N^2}$.
    - For example, $T = M = N^{2/3}$.
    - For $N = 2^{64}$: roughly 2 hours, 6 terabyte (with 1ns table lookup time...)

- Variants:
  - Distinguished points [Rivest 1982][Standaert Rouvroy Quisquater Legat 2002]
    - Reduces disk accesses from T to $\sqrt{T}$
  - Time/memory/data tradeoffs for stream ciphers [Biryukov Shamir 2000]
  - Rainbow tables: $\underline{2TM^2 = N^2}$ (but slightly longer table...) [Oeschlin 2003]
    - Use different functions in each iteration
    - Free Rainbow Tables http://www.freerainbowtables.com
      - MD5
      - SMB passwords (LM and NTLM)
    - Offer a 500GB disk with the MD5 rainbow table for US$400.
    - Distributed computation: chain-traversing client ran on volunteer's computer
  - Invert any function (no randomness assumption) [Fiat Naor 1991]
  - Lower bound of $T = \Omega\left(\frac{N^2}{M^2 \lg N}\right)$ for on "natural variants" [Barkan Biham Shamir 2006]

2 The rho method for finding collisions

- Collision Finding Problem: given access to $f : \{0, \ldots, N-1\} \to \{0, \ldots, N-1\}$, find $x, y \in \{0, \ldots, N-1\}$ such that $f(x) = f(y)$.

- Cryptanalytic applications:
  - Finding collisions in hash functions
  - Discrete logarithm problem (sort of)
  - Problem Set 2
- Collision finding via birthday paradox (time $\sqrt{N}$, space $\sqrt{N}$).
- Pollard's rho
  - "ρ" structure (Birthday paradox still holds)
  - Floyd's "two-finger" / "tortoise and hare" cycle finding algorithm
    * Let α be the leader and β be the cycle length.
    * Traverse $f^i(x_0)$ and $f^{2i}(x_0)$ concurrently.
    * When the sequences collide, $f^i(x_0) = f^{2i}(x_0)$, we have $i = \alpha+\gamma$ and $2i = \alpha+k\beta+\gamma$ for some k, γ. Thus $i = k\gamma$, a multiple of the cycle length.
    * Traverse $f^j(x_0)$ (starts at origin) and $f^{i+j}(x_0)$ (starts inside the cycle) concurently. When the sequences first collide, $f^j(x_0)$ has just entered the cycle and we have the collision in f.
- Variants
  - Leave "bread crumbs" (distinguished points) — improves constants
  - Brent's "binary search" algorithm — improves constants
  - Parallelized version [van Oorschot, Wiener 1996]

3 Massive cryptanalytic computations

- Exhaustive search
  - 56-bit DES broken in 1997
    * 1997: 96 days using ~14,000 volunteers (DESCHALL)
    * 1999: 22.5 hours
      · distributed.net: >100,000 voluntee
      · EFF DES Cracker: 36,864 custom-produced ASIC chips, <US$250K
    * 2006: 9 days US$10000 (COPACOBANA)
    * 2006: <1hr using a LAN Party's worth of PlayStations
  - 56-bit RC5 broken in 1997 (distributed.net)
  - 64-bit RC5 in 2002 (distributed net)
  - 72-bit RC5 challenge remains unbroken

- Hash function collisions
  - Structured MD5 collision: a PlayStation running for 20 days generating a rogue CA certificate
- Factoring (RSA)
  - Brute force: out of the question (key size $k \gg 100$). Best algorithm: Number Field Sieve with subexponential complexity $2^{(k^{1/3}(\log k)^{2/3}(1+o(1))}$.
  - Factoring records:

Year	Size of composite (bits)
1991	330
1994	426
1999	512
2003	576
2005	663
?	768

  - Breaking 1024-bit RSA using NFS on standard PCs estimated (until recently) to take $\sim 10^{12}$ US\$$\times$year (100M PCs with 170GB each)
    - Enshrined for many years to come in government standards and industry practice (e.g., SSL Certificate Authority keys trusted by your browser)
  - Special-purpose hardware
    - Bicycle chains
    - Opto-electronics (TWINKLE)
    - Massively-parallel custom chips (TWIRL, SHARK)
    - Currently: down to 1M US\$$\times$year (but: power, cooling, network, initial investment...)
    - See more at http://people.csail.mit.edu/tromer/cryptodev