Hash Function Combiners Can Be Secure Even When All The Hash Functions Are Weak

Yaakov Hoch and Adi Shamir

Department of Computer Science and Applied Mathematics
Weizmann Institute of Science

12/2/2008

Combiners in Cryptography

- Let f and g be two cryptographic primitives (encryption schemes, signature schemes, hash functions...)

Combiners in Cryptography

- Let f and g be two cryptographic primitives (encryption schemes, signature schemes, hash functions...)
- We would like to combine them into another primitive of the same type which will be:

Combiners in Cryptography

- Let ***f*** and ***g*** be two cryptographic primitives (encryption schemes, signature schemes, hash functions...)
- We would like to combine them into another primitive of the same type which will be:
- more secure than the separate ***f*** and ***g*** if both of them are secure

Combiners in Cryptography

- Let ***f*** and ***g*** be two cryptographic primitives (encryption schemes, signature schemes, hash functions...)
- We would like to combine them into another primitive of the same type which will be:
- more secure than the separate ***f*** and ***g*** if both of them are secure
- remain secure if at least one of ***f*** and ***g*** is secure, regardless of how the other primitive fails

Concatenated iterated hash functions

- Let f and g be MD iterated hash functions. Combine them by concatenating their outputs: $f(M) \circ g(M)$

Concatenated iterated hash functions

- Let f and g be MD iterated hash functions. Combine them by concatenating their outputs: $f(M) \circ g(M)$
- The output is twice as long, so if both f and g are strong we hope to make it harder to find collisions.

Concatenated iterated hash functions

- Let f and g be MD iterated hash functions. Combine them by concatenating their outputs: $f(M) \circ g(M)$
- The output is twice as long, so if both f and g are strong we hope to make it harder to find collisions.
- This was shown to be incorrect by Joux's multicollision attack in 2004.

Concatenated iterated hash functions

- Let f and g be MD iterated hash functions. Combine them by concatenating their outputs: $f(M) \circ g(M)$
- The output is twice as long, so if both f and g are strong we hope to make it harder to find collisions.
- This was shown to be incorrect by Joux's multicollision attack in 2004.

Joux's multicollision attack when one of the primitives is weak

- If f is weak, it may be easy to find a multicollision structure in f, but it will still require $2^{n/2}$ time to find a collision in the strong g with these messages.

Joux's multicollision attack when one of the primitives is weak

- If f is weak, it may be easy to find a multicollision structure in f, but it will still require $2^{n/2}$ time to find a collision in the strong g with these messages.
- If f is strong and g is weak, we can reverse the roles of f and g, but the attack will still require $2^{n/2}$ time.

Joux's open problem

- Can you find a better attack on concatenated iterated hash functions if it is easy to find collisions in both f and g?

Joux's open problem

- Can you find a better attack on concatenated iterated hash functions if it is easy to find collisions in both f and g?
- The first issue: How to define a meaningful model in which f and g are weak only in this sense, and not for example, in mapping every input to a constant output.

Joux's open problem

- Can you find a better attack on concatenated iterated hash functions if it is easy to find collisions in both f and g?
- The first issue: How to define a meaningful model in which f and g are weak only in this sense, and not for example, in mapping every input to a constant output.

Our Very Strong Adversarial Model:

- f and g are functions from the previous chaining value h and the message block m to the next chaining value h'.

Our Very Strong Adversarial Model:

- f and g are functions from the previous chaining value h and the message block m to the next chaining value h'.
- The attacker is given access to the following six random oracles, which enable him to find for any two values a corresponding third value:
- Forward query: $f(h, m, ?)$, $g(h, m, ?)$
- Backward query: $f(?, m, h')$, $g(?, m, h')$
- Bridging query: $f(h, ?, h')$, $g(h, ?, h')$

Examples of what the attacker can do in linear time:

- Find collisions, multicollisions, and collision trees
- Find a self loop mapping IV to itself
- Find expandable messages
- Find diamonds and kites
- connect chaining values created by f with chaining values created by g

Our main result:

- Theorem: $f(M) \oplus g(M)$ is indifferentiable from a random oracle using fewer than $2^{n/2}$ queries.

Our main result:

- Theorem: $f(M) \oplus g(M)$ is indifferentiable from a random oracle using fewer than $2^{n/2}$ queries.
- Corollary: Since a collision in $f(M) \circ g(M)$ implies a collision in $f(M) \oplus g(M)$, but finding collisions in a random oracle with n-bit outputs requires $O(2^{n/2})$ time, the attacker cannot find a faster generic attack against concatenated iterated hash functions even when both f and g are weak, provided that they are sufficiently random and sufficiently independent.