Iterated Random Oracle: A Universal Approach for Finding Loss in Security Reduction

Fuchun Guo, Willy Susilo, Yi Mu, Rongmao Chen,
Jianchang Lai, Guomin Yang

UNIVERSITY OF WOLLONGONG AUSTRALIA

國防科學技術大學
National University of Defense Technology

About this work

- In the random oracle model, we can prove a cryptosystem such as a public-key encryption in the IND (indistinguishability) security model under a computational hard problem, e.g. CDH.

About this work

- In the random oracle model, we can prove a cryptosystem such as a public-key encryption in the IND (indistinguishability) security model under a computational hard problem, e.g. CDH.
- The solution to the computational hard problem comes from one of hash queries to the random oracle made by the adversary.

About this work

- In the random oracle model, we can prove a cryptosystem such as a public-key encryption in the IND (indistinguishability) security model under a computational hard problem, e.g. CDH.
- The solution to the computational hard problem comes from one of hash queries to the random oracle made by the adversary.
- When the decisional variant of this problem is also hard, the simulator does not know which query contains the correct solution.

About this work

- In the random oracle model, we can prove a cryptosystem such as a public-key encryption in the IND (indistinguishability) security model under a computational hard problem, e.g. CDH.
- The solution to the computational hard problem comes from one of hash queries to the random oracle made by the adversary.
- When the decisional variant of this problem is also hard, the simulator does not know which query contains the correct solution.
- Finding loss refers to finding an incorrect solution from queries.

About this work

- In the random oracle model, we can prove a cryptosystem such as a public-key encryption in the IND (indistinguishability) security model under a computational hard problem, e.g. CDH.
- The solution to the computational hard problem comes from one of hash queries to the random oracle made by the adversary.
- When the decisional variant of this problem is also hard, the simulator does not know which query contains the correct solution.
- Finding loss refers to finding an incorrect solution from queries.
- We introduce Iterated random oracle (a complex random oracle) to address the finding loss towards tight(er) reduction.

Two Types of Security Reductions

In a security reduction, a simulator uses an adversary's attack to solve a hard problem. There are two types of reductions.

Two Types of Security Reductions

In a security reduction, a simulator uses an adversary's attack to solve a hard problem. There are two types of reductions.

- Unforgeability security based on a computational hard problem (UF-CHP). For example, in a digital signature scheme, the simulator uses the forged signature to solve a computational hard problem.

Two Types of Security Reductions

In a security reduction, a simulator uses an adversary's attack to solve a hard problem. There are two types of reductions.

- Unforgeability security based on a computational hard problem (UF-CHP). For example, in a digital signature scheme, the simulator uses the forged signature to solve a computational hard problem.
- Indistinguishability security based on a decisional hard problem (IND-DHP). For example, in a public-key encryption scheme, the simulator uses the guess of the random message in CT to solve a decisional hard problem.

IND-Computational Hard Problem

IND security based on a computational hard problem (IND-CHP) ???

UNIVERSITY
OF WOLLONGONG
AUSTRALIA

IND-Computational Hard Problem

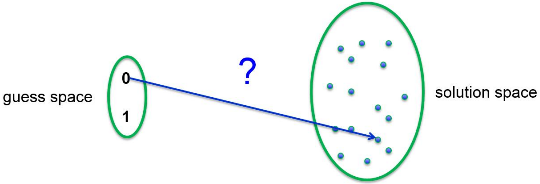
IND security based on a computational hard problem (IND-CHP) ???

In this security model and reduction:
The adversary's output: $\{0, 1\}$
The simulator's output : solution to a computational hard problem.

IND-Computational Hard Problem

IND security based on a computational hard problem (IND-CHP) ???

In this security model and reduction:
The adversary's output: $\{0, 1\}$
The simulator's output : solution to a computational hard problem.



It seems impossible to carry out such a security reduction because the guess 0 or 1 cannot provide sufficient information to find a correct solution from an exponential-size solution space.

UNIVERSITY
OF WOLLONGONG
AUSTRALIA

IND-CHP in Random Oracles

However, using random oracles [BR93], IND-CHP reduction is possible!

Suppose a hash function H is treated as a random oracle. In the random oracle model, when the adversary makes a query on a string x to the random oracle:

- $H(x)$ is uniformly random and independent of x.
- $H(x)$ is controlled by the simulator (tricky part).

[BR93] *Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: Denning, D.E., Pyle, R., Ganesan, R., Sandhu, R.S., Ashby, V. (eds.) CCS 1993. pp. 62–73. ACM (1993)*

Core of Encryption Reduction in Random Oracles

Considering the following ciphertext:

$$CT = \left(g^x,\ g^y,\ H\left(g^{xy}\right) \oplus m_{coin}\right)$$

[1] assumption

UNIVERSITY OF WOLLONGONG AUSTRALIA

Core of Encryption Reduction in Random Oracles

Considering the following ciphertext:

$$CT = \left(g^x,\ g^y,\ H\left(g^{xy}\right) \oplus m_{coin}\right)$$

Suppose[1] an adversary $\mathcal{A}$ can distinguish the encrypted message $m_{coin} \in \{m_0, m_1\}$ in the random oracle model. We can construct a simulator to solve the CDH problem. Given $\left(g, g^a, g^b\right)$, the simulator aims to compute g^{ab} .

[1] assumption

UNIVERSITY OF WOLLONGONG AUSTRALIA

Core of Encryption Reduction in Random Oracles

Considering the following ciphertext:

$$CT = \left(g^x,\ g^y,\ H\left(g^{xy}\right) \oplus m_{coin}\right)$$

Suppose[1] an adversary $\mathcal{A}$ can distinguish the encrypted message $m_{coin} \in \{m_0, m_1\}$ in the random oracle model. We can construct a simulator to solve the CDH problem. Given $\left(g, g^a, g^b\right)$, the simulator aims to compute g^{ab} .

Simulation: $CT = (g^a,\ g^b,\ R)$, where R is a random string.

[1] assumption

UNIVERSITY OF WOLLONGONG AUSTRALIA

Core of Encryption Reduction in Random Oracles

Considering the following ciphertext:

$$CT = \left(g^x,\ g^y,\ H\left(g^{xy}\right) \oplus m_{coin}\right)$$

Suppose[1] an adversary $\mathcal{A}$ can distinguish the encrypted message $m_{coin} \in \{m_0, m_1\}$ in the random oracle model. We can construct a simulator to solve the CDH problem. Given $\left(g, g^a, g^b\right)$, the simulator aims to compute g^{ab} .

Simulation: $CT = (g^a,\ g^b,\ R)$, where R is a random string.

- No query on g^{ab}, no break on the ciphertext. (One-Time Pad)
- According to the assumption, g^{ab} will appear in one of queries.
- One of hash queries is the solution to the CDH problem.

---

[1] assumption

UNIVERSITY OF WOLLONGONG AUSTRALIA

Finding Loss

Suppose $\mathcal{A}$ made the following queries to the random oracle.

$$Q_1,\ Q_2,\ Q_3, \cdots, Q_q$$

Which Q is equal to g^{ab} ?

Finding Loss

Suppose $\mathcal{A}$ made the following queries to the random oracle.

$$Q_1,\ Q_2,\ Q_3, \cdots, Q_q$$

Which Q is equal to g^{ab} ?

- If the DDH problem is **easy**, the simulator can test each query until find the correct solution. The success probability of finding the correct solution is 1.

Finding Loss

Suppose $\mathcal{A}$ made the following queries to the random oracle.

$$Q_1,\ Q_2,\ Q_3,\cdots,Q_q$$

Which Q is equal to g^{ab} ?

- If the DDH problem is **easy**, the simulator can test each query until find the correct solution. The success probability of finding the correct solution is 1.
- If the DDH problem is also **hard**, the simulator has to randomly pick one query as the solution. The success probability of finding the correct solution is $1/q$.

Finding Loss

Suppose $\mathcal{A}$ made the following queries to the random oracle.

$$Q_1,\ Q_2,\ Q_3, \cdots, Q_q$$

Which Q is equal to g^{ab} ?

- If the DDH problem is **easy**, the simulator can test each query until find the correct solution. The success probability of finding the correct solution is 1.
- If the DDH problem is also **hard**, the simulator has to randomly pick one query as the solution. The success probability of finding the correct solution is $1/q$.

The number of hash queries q could be as large as 2^{60}.

Loose Reduction!

UNIVERSITY OF WOLLONGONG AUSTRALIA

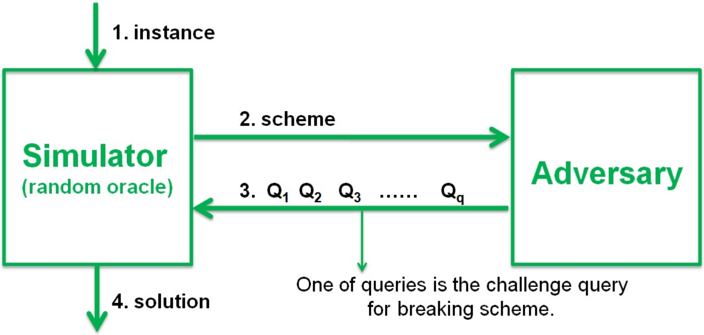
Finding Loss

How to find the correct solution from the adversary's query set?

We call this problem as a **finding problem** and the reduction has a **finding loss**, if the probability of finding the correct solution is < 1.

In this work, we focus on the non-trivial case that the decisional variant of a computational hard problem is also hard.

Security Reduction in IND-CHP



The simulator uses the query set to find the solution to the instance.

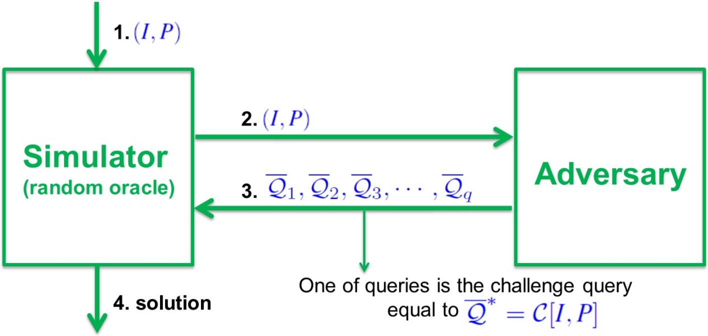
Security Reduction in IND-CHP

Let $\mathcal{C}[I,P]$ be a solution to an instance I under a computational hard problem P.

	Before Disclosing Simulation	After Disclosing Simulation
$\mathcal{A}$ is given	Scheme	Instance
$\mathcal{A}$ queries	A query set including a challenge query for breaking scheme	A query set including a challenge query equal to the solution

UNIVERSITY OF WOLLONGONG AUSTRALIA

Theory 1 (Traditional Approach)



The simulator can only solve the hard problem with success probability $\frac{1}{q}$.

Cash-Kiltz-Shoup Approach

- In EUROCRYPT 2008, Cash, Kiltz and Shoup [CKS08] proposed a new computational problem called the twin Diffie-Hellman problem.
- The new hard problem is as hard as the CDH problem, where the DDH problem is also hard.
- Schemes based on the twin Diffie-Hellman problem have no finding loss in security reduction.

[CKS08] *Cash, D., Kiltz, E., Shoup, V.: The twin diffie-hellman problem and applications. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 127–145. Springer,Heidelberg (2008).*
[CKS09] *Cash, D., Kiltz, E., Shoup, V.: The twin diffie-hellman problem and applications. J. Cryptology 22(4), 470–504 (2009).*

Trapdoor Test in Cash-Kiltz-Shoup Approach

Given an instance I_1, suppose there exist a particularly constructed instance I_2 and a trapdoor test algorithm such that:

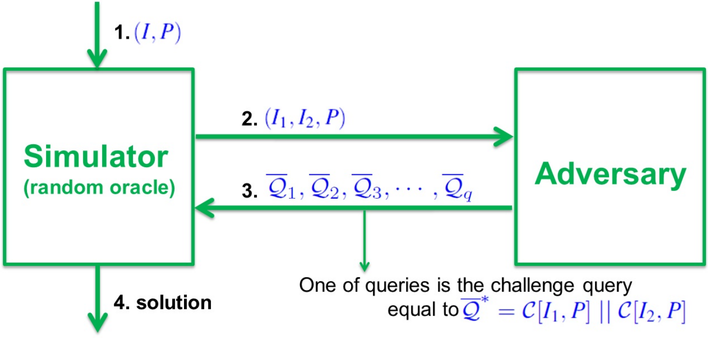
TrapdoorTest(Q_1, Q_2)=True if and only if

$$Q_1 = \mathcal{C}[I_1, P], \;\; Q_2 = \mathcal{C}[I_2, P],$$

except with a negligible probability.

UNIVERSITY OF WOLLONGONG AUSTRALIA

Theory 2 (Cash-Kiltz-Shoup)



The simulator can solve the hard problem with success probability 1 if there exists a trapdoor test on solutions to a given instance $I_1 (= I)$ and a created instance I_2.

Theory 2 (Cash-Kiltz-Shoup)

Summary:

- Cash-Kiltz-Shoup approach is smart and easy in understanding.
- This approach requires a trapdoor test.
- The proposed trapdoor test can be adopted by some computational Diffie-Hellman hard problems only. (Limitation & Our Motivation)

What is Iterated Random Oracle?

Suppose the adversary needs to make a challenge query in order to use its output to break a scheme.

What is Iterated Random Oracle?

Suppose the adversary needs to make a challenge query in order to use its output to break a scheme.

1. Traditional Random Oracle (one special input)

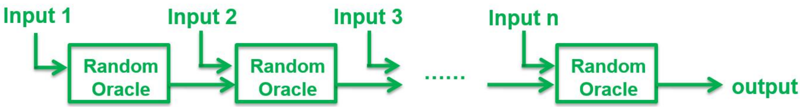
UNIVERSITY
OF WOLLONGONG
AUSTRALIA

What is Iterated Random Oracle?

Suppose the adversary needs to make a challenge query in order to use its output to break a scheme.

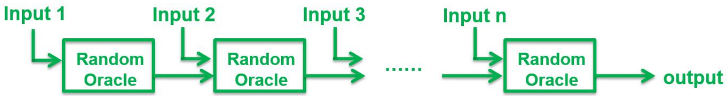
1. Traditional Random Oracle (one special input)

2. Iterated Random Oracle (n special inputs)



UNIVERSITY OF WOLLONGONG AUSTRALIA

Iterated Query in the Iterated Random Oracle

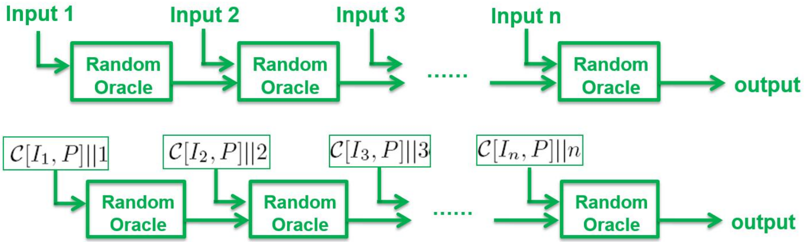


Iterated Query. We define an iterated query $\overline{\mathcal{Q}}$ to the random oracle as

$$\overline{\mathcal{Q}} = \text{Response} \mid\mid \text{Weight} \mid\mid \text{Iteration Time} = \overline{\mathcal{R}} \mid\mid Q \mid\mid i,$$

$\overline{\mathcal{R}}$: a response of a hash query or an empty string 0_ϵ,
Q: a weight (any arbitrary string) chosen by the adversary,
i: the iteration time.

Challenge Query in Iterated Random Oracle

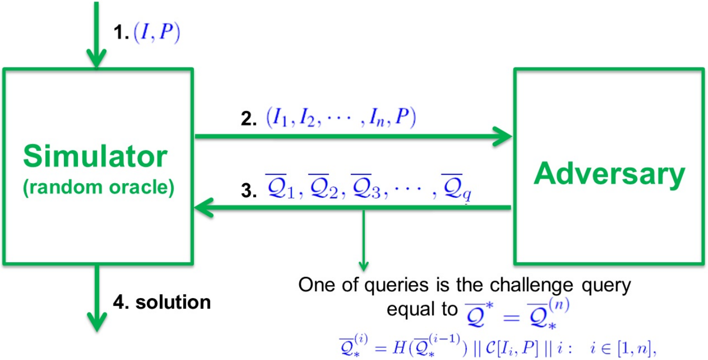


$$\overline{\mathcal{Q}}_*^{(i)} = H(\overline{\mathcal{Q}}_*^{(i-1)})||\mathcal{C}[I_i, P] \; || \; i : \; i \in [1, n],$$

where $H(\overline{\mathcal{Q}}_*^{(0)}) = 0_\epsilon$ is an empty string.

$\overline{\mathcal{Q}}_*^{(n)}$ is the defined challenge query.

Theory 3 (Iterated Random Oracle)



The simulator can solve the hard problem
with success probability $\frac{1}{nq^{\frac{1}{n}}}$.

UNIVERSITY OF WOLLONGONG AUSTRALIA

Comparison of Three Theories

	Theory 1 (Traditional)	Theory 2 (CKS)	Theory 3 (Ours)
For All Problems	✓	×	✓
Success Probability	$\frac{1}{q}$	1	$\frac{1}{n \cdot q^{\frac{1}{n}}}$
Finding Efficiency	$O(1)$	$O(q)$	$O(n)$
Query Efficiency	1	2	$O(n)$

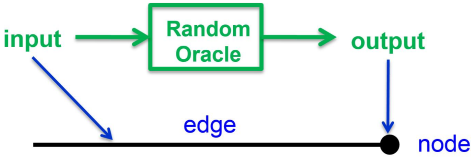
Table : Comparison of success probability.

	$q = 2^{40}$	$q = 2^{50}$	$q = 2^{60}$
Traditional Approach	$\frac{1}{2^{40}}$	$\frac{1}{2^{50}}$	$\frac{1}{2^{60}}$
Cash-Kiltz-Shoup	1	1	1
Iterated Random Oracle with $n = 10$	$\frac{1}{160}$	$\frac{1}{320}$	$\frac{1}{640}$

UNIVERSITY OF WOLLONGONG AUSTRALIA

Queries and Tree Representation

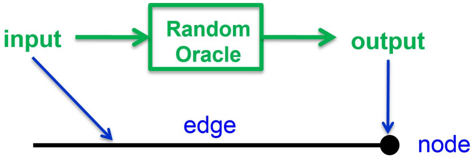
All queries and responses are represented using a tree.



UNIVERSITY
OF WOLLONGONG
AUSTRALIA

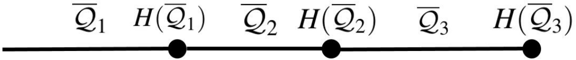
Queries and Tree Representation

All queries and responses are represented using a tree.



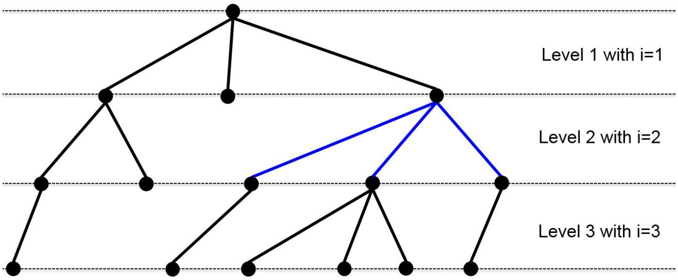
For example:

$$\overline{Q}_1 = 0_\epsilon||Q_1||1, \qquad \overline{Q}_2 = H(\overline{Q}_1)||Q_2||2, \qquad \overline{Q}_3 = H(\overline{Q}_2)||Q_3||3$$



UNIVERSITY OF WOLLONGONG AUSTRALIA

Properties of Tree Representation

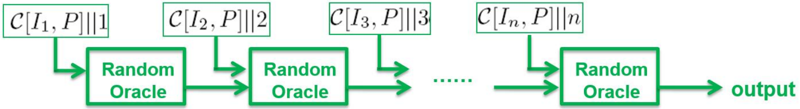


$$\overline{Q} = \text{Response} \,||\, \text{Weight} \,||\, \text{Iteration Time} = \overline{\mathcal{R}} \,||\, Q \,||\, i,$$

- All queries with the same iteration time i are edges at the level i.
- All queries with the same response are edges from the same node.
- All edges starting from the same node must have different weights.

UNIVERSITY OF WOLLONGONG AUSTRALIA

Properties of Tree Representation

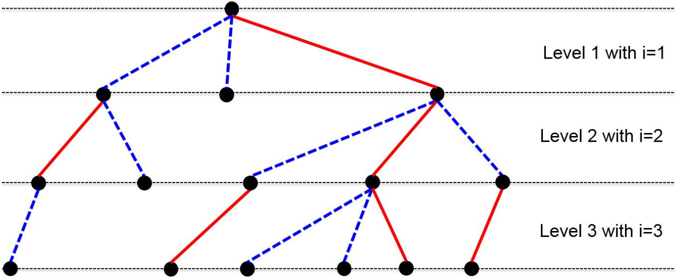


$P = CDH,$ $\quad I_i = (g, g^{a_i}, g^b)$ $\quad \mathcal{C}[I_i, P] = g^{a_i b}$

Red & Solid edge at level i denotes a query with a valid weight $= g^{a_i b}$

Blue & Dashed edge at level i denotes a query with an invalid weight $\neq g^{a_i b}$

Properties of Tree Representation



- Each level could have more than one red & solid edge.
- All red & solid edges at the same level must be from different nodes.
- There exists one red & solid path from the root to a leaf $H(\overline{Q}^*)$.

Proof of Our Theory

Simulator Construction. Given (I, P), the simulator works as follows.

- Randomly choose $d \in [1, n]$ and set $I_d = I$.
- Choose random instances $I_1, I_2, \cdots, I_{d-1}, I_{d+1}, \cdots, I_n$ such that $\mathcal{C}[I_i, P]$ for all $i \in [1, n] \setminus \{d\}$ are known by the simulator.

Each instance should be indistinguishable such that d is unknown to the adversary (very important!).

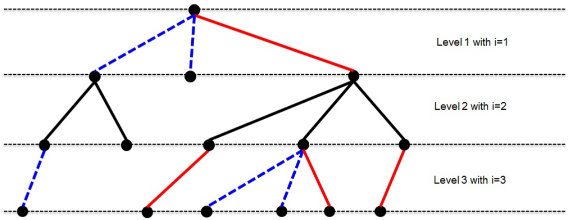
Proof of Our Theory

- $\mathcal{C}[I_d, P] = \mathcal{C}[I, P]$ is unknown.
- $\mathcal{C}[I_i, P]$ for all $i \in [1, n] \setminus \{d\}$ are known.

1. The solution will appear in one of edges at the d-th level.
2. Use known solutions at levels $d + 1$ to n to filter useless queries.
3. Randomly pick a query from candidate queries as a valid query.

Proof of Our Theory

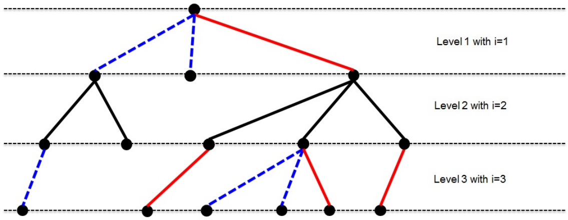
- The query $\overline{Q}$ at the level i is a **valid query** if its weight is $g^{a_i b}$.
- The query $\overline{Q}$ is a **candidate query** if there exists a red $\&$ solid path from the node $H(\overline{Q})$ to a leaf node at the level n. All queries at the level n are candidate queries.
- The query $\overline{Q}$ is a **useless query** if there exists no red $\&$ solid path from the node $H(\overline{Q})$ to a leaf node at the level n.



In the above example, $d = 2$. The simulator does not know whether a query at the level 2 is a valid query or not, but knows.....

Proof of Our Theory

	Randmly choose a query from
Traditional Approach	all queries
Iterated Random Oracle	candidate queries at the level d



Proof of Our Theory

1. (Lemma 1) If the following rate

$$R^{(i)} = \frac{\text{The number of valid queries in } \mathbb{Q}^{(i)}}{\text{The number of candidate queries in } \mathbb{Q}^{(i)}} < \frac{1}{q^{\frac{1}{n}}}$$

holds for all $i \in [1, n]$, the adversary must make more than q queries.

UNIVERSITY OF WOLLONGONG AUSTRALIA

Proof of Our Theory

1. (Lemma 1) If the following rate

$$R^{(i)} = \frac{\text{The number of valid queries in } \mathbb{Q}^{(i)}}{\text{The number of candidate queries in } \mathbb{Q}^{(i)}} < \frac{1}{q^{\frac{1}{n}}}$$

holds for all $i \in [1, n]$, the adversary must make more than q queries.

2. For q hash queries at most, there must exist an $i^* \in [1, n]$ such that

$$R^{(i^*)} = \frac{\text{The number of valid queries in } \mathbb{Q}^{(i^*)}}{\text{The number of candidate queries in } \mathbb{Q}^{(i^*)}} \geq \frac{1}{q^{\frac{1}{n}}}.$$

UNIVERSITY OF WOLLONGONG AUSTRALIA

Proof of Our Theory

1. (Lemma 1) If the following rate

$$R^{(i)} = \frac{\text{The number of valid queries in } \mathbb{Q}^{(i)}}{\text{The number of candidate queries in } \mathbb{Q}^{(i)}} < \frac{1}{q^{\frac{1}{n}}}$$

holds for all $i \in [1, n]$, the adversary must make more than q queries.

2. For q hash queries at most, there must exist an $i^* \in [1, n]$ such that

$$R^{(i^*)} = \frac{\text{The number of valid queries in } \mathbb{Q}^{(i^*)}}{\text{The number of candidate queries in } \mathbb{Q}^{(i^*)}} \geq \frac{1}{q^{\frac{1}{n}}}.$$

3. When $d = i^*$,

$$\begin{aligned} \Pr[suc] &= \sum_{i=1}^{n} \Pr[suc|d = i] \Pr[d = i] \\ &\geq \Pr[suc|d = i^*] \Pr[d = i^*] = \frac{1}{n} \cdot \frac{1}{q^{\frac{1}{n}}} \end{aligned}$$

Proof of Our Theory

Examples: $n = 2, q = 8$.

The probability should be at least $\frac{1}{nq^{\frac{1}{n}}} = \frac{1}{2\sqrt{8}}$.

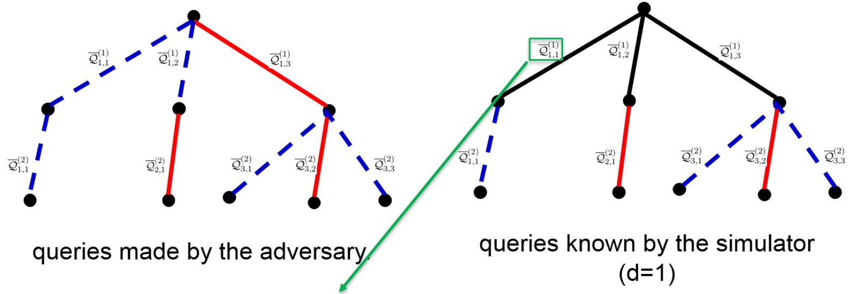
The probability $\Pr[suc|d = i^*]$ for some i^* should be at least $\frac{1}{\sqrt{8}}$.

Proof of Our Theory

Examples: $n = 2, q = 8$.

The probability should be at least $\frac{1}{nq^{\frac{1}{n}}} = \frac{1}{2\sqrt{8}}$.

The probability $\Pr[suc|d = i^*]$ for some i^* should be at least $\frac{1}{\sqrt{8}}$.

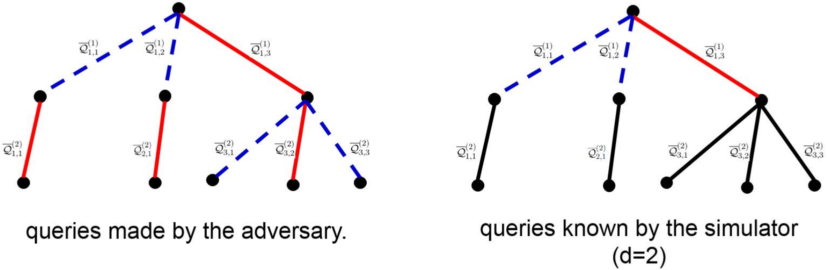


When $d = 1$, the query $\overline{Q}^{(1)}_{1,1}$ will be removed because of no red & solid path. Therefore, we have $\Pr[suc|d = 1] = \frac{1}{2} \geq \frac{1}{\sqrt{8}}$.

UNIVERSITY OF WOLLONGONG AUSTRALIA

Proof of Our Theory

Examples: $n = 2, q = 8$.
The probability should be at least $\frac{1}{nq^{\frac{1}{n}}} = \frac{1}{2\sqrt{8}}$.
The probability $\Pr[suc|d = i^*]$ for some i^* should be at least $\frac{1}{\sqrt{8}}$.



When $d = 2$, it is easy to see that $\Pr[suc|d = 2] = \frac{3}{5} \geq \frac{1}{\sqrt{8}}$.

Theories in Applications

Theories	Instance(s)	Challenge Query
Traditional Approach	I	$\overline{\mathcal{Q}}^* = \mathcal{C}[I, P]$
Cash-Kiltz-Shoup	(I_1, I_2)	$\overline{\mathcal{Q}}^* = \mathcal{C}[I_1, P] \,\|\|\, \mathcal{C}[I_2, P]$
Iterated Random Oracle	$(I_1, I_2, \cdots, I_n)$	$\overline{\mathcal{Q}}^* = \overline{\mathcal{Q}}_*^{(n)}$

To apply the theories:

- The scheme must be simulated using the generated instance(s).
- The defined challenge query must be made to break the scheme.

Applications

- Generic conversion for Key Encapsulation Mechanism (KEM):

  One-Way KEM to IND-KEM with a small finding loss in the random oracle mode without expending ciphertext size.

- Tight reduction for Key Exchange under the IND-CHP reduction.

Advantage: tighter reduction with a small finding loss

Disadvantage: Longer private/secret key (linear n, $n = 10$)

Conclusion

- Introduced the finding loss in the IND-CHP reduction.
- Proposed iterated random oracle to reduce the finding loss.

	Theory 1 (Traditional)	Theory 2 (CKS)	Theory 3 (Ours)
For All Problems	✓	×	✓
Success Probability	$\frac{1}{q}$	1	$\frac{1}{n \cdot q^{\frac{1}{n}}}$
Finding Efficiency	$O(1)$	$O(q)$	$O(n)$
Query Efficiency	1	2	$O(n)$

- Showed applications in encryption and key exchange.

Thanks & Questions