On the (In)Security of the Diffie-Hellman Oblivious PRF with Multiplicative Blinding

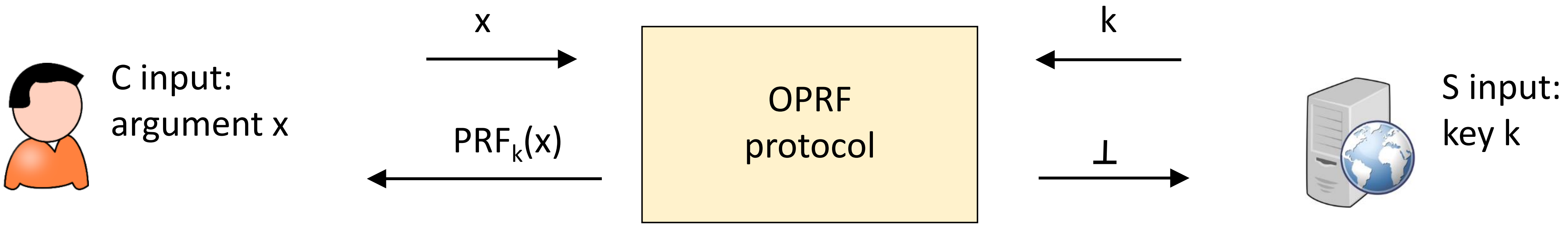
Stanislaw Jarecki

Hugo Krawczyk

Jiayu Xu

Oblivious Pseudorandom Function (OPRF) [NR'97,FIPR'05]

Assume $\mathrm{PRF}_k(x)$ is a PRF with key k and argument x



- Application Examples #1: "Password-hardening" [FK'00, Boyen'09, JKKX'16, JKX'18, …]:
    - $\mathrm{OPRF}_k(\cdot)$ maps low-entropy secrets to pseudorandom keys
    - OPAQUE [JKX'18]: $\mathrm{OPRF}_k(\cdot)$ can upgrade AKE to strong asymmetric PAKE
- Application Examples #2: Set Intersection [HL'08, …]:
    - Alice runs $\mathrm{OPRF}_k(\cdot)$ so Charlie computes: $\{ \mathrm{PRF}_k(x) \text{ for } x \in \mathrm{Set}_{Charlie} \}$
    - Alice sends to Charlie: $\{ \mathrm{PRF}_k(y) \text{ for } y \in \mathrm{Set}_{Alice} \}$
- Many other applications: SSE [JJKRS'13], Two-Factor Auth [JHSS'18], *Privacy Pass* [DGSTV'18], Key Management [JKR'19], Anonymous Tickets [KDMT'20], Contact Tracing [DPT'20], …

UC OPRF: *Exponential-blinded* Hashed Diffie-Hellman [...,JKKX'16]

- H hashes onto a group (order p)
- PRF key $k \leftarrow Z_p$

Hashed DH PRF

$$PRF_k(x) \triangleq H'(x, H(x)^k)$$

C input:
argument x

$r \leftarrow Z_p$

$a = H(x)^r$ →

← $b = a^k$

$PRF_k(x) = H'(x, b^{1/r})$

S input:
key k

- outer hash H' de-correlates PRF outputs on correlated keys, e.g., PRF_k and PRF_{2k}
- adding x to hash H' inputs disambiguates H' query as PRF evaluation on a unique (key,argument) pair

Note: $b^{1/r} = (a^k)^{1/r} = ((H(x)^r)^k)^{1/r} = H(x)^k$

- Protocol uses **1** exp for S and **2** exps for C
- [JKKX'16]: realizes Universally Composable OPRF under (Gap) One-More DH in ROM
- Question: Can this OPRF be implemented even faster?

OPRF candidate: *Multiplicative-blinded* Hashed DH

Hashed DH PRF

$$\mathrm{PRF}_k(x) \triangleq H'(\,x, H(x)^k\,)$$

C input:
argument x

$r \leftarrow Z_p$

$a = H(x) \cdot g^r$ →

← $b = a^k\ ,\ \ z = g^k$

$\mathrm{PRF}_k(x) = H'(x\,, b{\cdot}z^{-r})$

S input:
key k
(+ "public key" $z = g^k$)

$$b \cdot z^{-r} = a^k \cdot z^{-r}$$
$$= (H(x) \cdot g^r)^k \cdot (g^k)^{-r}$$
$$= H(x)^k$$

- *multiplicative* instead of *exponential* blinding (≈ Chaum's Blind RSA scheme)
- C replaces 2 ***var-base*** exps with 2 ***fixed-base*** exps (or 1fb+1vb if PRF key z new)
- up to ~ 6-7x speedup for 128-bit security with precomputation
- Question: Is mult-blinded HDH just as secure as exp-blinded HDH?

Multiplicative-blinded Hashed DH: server-side attack

recall what we wanted to compute:

Hashed DH PRF

$\mathrm{PRF}_k(x) \triangleq H'(x, H(x)^k)$

malicious S* implements $\mathrm{OPRF}_{k,\delta}(\cdot)$ for new a PRF:

Effective PRF for malicious server

$\mathrm{PRF}_{k,\delta}(x) \triangleq H'(x, \delta \cdot H(x)^k)$

C input:
argument x

$r \leftarrow Z_p$

$a = H(x) \cdot g^r$ →

← $b = \delta \cdot a^k$, $z = g^k$

$H'(x, b \cdot z^{-r}) = H'(x, \delta \cdot H(x)^k)$
$= \mathrm{PRF}_{k,\delta}(x)$

S* input:
key k
+ shift value δ

$b \cdot z^{-r} = \delta \cdot a^k \cdot z^{-r}$
$= \delta \cdot (H(x) \cdot g^r)^k \cdot (g^k)^{-r}$
$= \delta \cdot H(x)^k$

<u>Main Question:</u> Is $\mathrm{PRF}_{k,\delta}(\cdot)$ substantially different from $\mathrm{PRF}_k(\cdot)$?

- Functions $\mathrm{PRF}_k(\cdot)$ are ≈ independent RF's for any keys k chosen by S* (by UC OPRF [JKKX'16])
- Functions $\mathrm{PRF}_{k,\delta}(\cdot)$ can have programmed collisions (="correlated outputs") for keys (k,δ) chosen by S*

For any (k,δ) and any x^*, set (k^*,δ^*) s.t. $\delta^* \cdot H(x^*)^{k^*} = \delta \cdot H(x^*)^k$

1. $\mathrm{PRF}_{k^*,\delta^*}(x) = \mathrm{PRF}_{k,\delta}(x)$ if $x = x^*$
2. $\mathrm{PRF}_{k^*,\delta^*}(x) \neq \mathrm{PRF}_{k,\delta}(x)$ if $x \neq x^*$ [≈ independent functions in ROM (this paper)]

Server-side attack on *mult-blinded* HDH OPRF (app. example)

In *multiplicative-blinded* Hashed DH, malicious S* can pick (k,δ), (k^*,δ^*) and x^* s.t.

1.	$PRF_{k^*,\delta^*}(x) = PRF_{k,\delta}(x)$	if $x = x^*$	
2.	$PRF_{k^*,\delta^*}(x) \neq PRF_{k,\delta}(x)$	if $x \neq x^*$	[≈ independent functions in ROM]

Charlie on Monday

x = pwd →

$v = PRF_{k,\delta}(x)$ ←

$OPRF_{k,\delta}(\cdot)$

← (k,δ)

⊥ →

attacker S*

uses v as encryption key,
stores c = AuthEnc(v,data)

Charlie on Tuesday

x = pwd →

$v' = PRF_{k^*,\delta^*}(x)$ ←

$OPRF_{k^*,\delta^*}(\cdot)$

← (k*,δ*)

⊥ →

uses v’ to decrypt c,
succeeds iff v’=v

protests/complains/retries
iff decryption fails!

bit b = (v’ == v) = (x == x*) →

S* learns if Charlie’s pwd = x*

No such attack on UC OPRF [JKKX'16], i.e. *exp-blinded* HDH

In *exponential-blinded* Hashed DH, functions $PRF_k(x)$ are independent for all keys k, k^*:

1. $PRF_{k*}(x) = PRF_k(x)$ for all x if $k^* = k$
2. $PRF_{k*}(x) \neq PRF_k(x)$ for all x if $k^* \neq k$ [≈ independent functions in ROM]

Charlie on Monday

$x = pwd$ →

← $v = PRF_k(x)$

$OPRF_k(\cdot)$

← k

⊥ →

attacker S*

uses v as encryption key,
stores c = AuthEnc(v,data)

Charlie on Tuesday

$x = pwd$ →

← $v' = PRF_{k*}(x)$

$OPRF_{k*}(\cdot)$

← k*

⊥ →

uses v' to decrypt c,
succeeds iff v'=v

protests/complains/retries
iff decryption fails!

bit b = (v' == v) = (k == k*) →

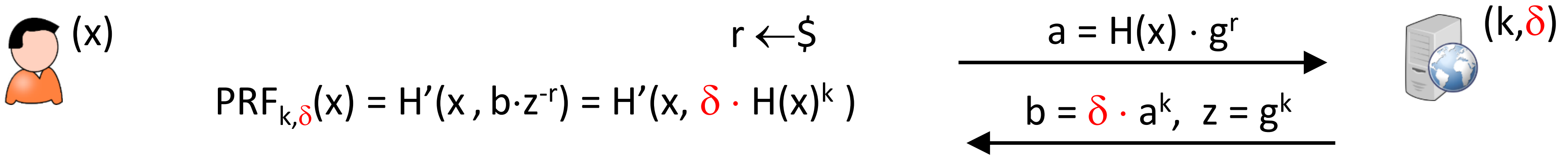
with exp-blinded HDH
S* doesn't learn
anything new!

Multiplicative-blinded Hashed DH: Can we stop the attack?

Effective PRF for malicious server

$$PRF_{k,\delta}(x) \triangleq H'(\,x, \delta \cdot H(x)^k\,)$$

Attack: S correlates $PRF_{k,\delta}$ functions on chosen arguments x**



How to remove δ, or remove its effects on adversarial ability to correlate $PRF_{k,\delta}$ functions?

1. S sends NIZKP that $\exists k$ s.t. $(b,z) = (a^k, g^k)$ — bandwidth++, adds 1-2 exp's for C and S

   ⇒ if you need *Verifiable* OPRF then use mult-blind and NIZKP, otherwise do exp-blind

2. C verifies the server's public key z — requires certificates, not good for e.g. PAKEs

   (S can still δ-shift C's hash calculation but the shift is not x-dependent, so no correlations)

3. Modify PRF to: $PRF_k(x) \triangleq H'(\,x, g^k, H(x)^k\,)$ — does it come at no cost?

Pros and Cons of adding g^k to the hash in Hashed DH OPRF

	Hashed DH PRF $PRF_k(x) \triangleq H'(x, H(x)^k)$	Modified Hashed DH PRF $PRF_k(x) \triangleq H'(x, g^k, H(x)^k)$
Mult-blinded oblivious evaluation (x) → $a = H(x) \cdot g^r$ → (k) (x) ← $b = a^k$, $z = g^k$ ← (k) $H'(x, b \cdot z^{-r})$ vs. $H'(x, g^k, b \cdot z^{-r})$	Security? • breaks some applications ✘ • This paper: It is secure for some important applications (with disclaimers) ✓	Realizes UC OPRF of [JKKX'16] • *fewer exp's than Exp-blinded* ✓
Exp-blinded oblivious evaluation (x) → $a = H(x)^r$ → (k) (x) ← $b = a^k$, $z = g^k$ ← (k) $H'(x, b^{(1/r)})$ vs. $H'(x, g^k, b^{(1/r)})$	Realizes UC OPRF of [JKKX'16] ✓ The no-g^k version allows e.g. bandwidth-restricted devices to use Exp-blinded evaluation	Realizes UC OPRF of [JKKX'16] ✓ • must store/send $z=g^k$ • IRTF CFRG: push-back from e.g. IOT applications ✘

Definitional Approach: Correlated UC OPRF (this paper)

Hashed DH PRF

$$PRF_k(x) \triangleq H'(\, x, H(x)^k\,)$$

Exp-blinded oblivious evaluation

Effective PRF for malicious servers

$$PRF_k(x) \triangleq H'(\, x, H(x)^k\,)$$

"Strong" UC OPRF [JKKX'16]

- PRF_k and PRF_{k*} $\approx$ indep. RF's $\forall k^* \neq k$
- realized by exp-blinded Hashed-DH (under Gap OneMore-DH in ROM)

Mult-blinded oblivious evaluation

Effective PRF for malicious servers

$$PRF_{k,\delta}(x) \triangleq H'(\, x, \delta \cdot H(x)^k\,)$$

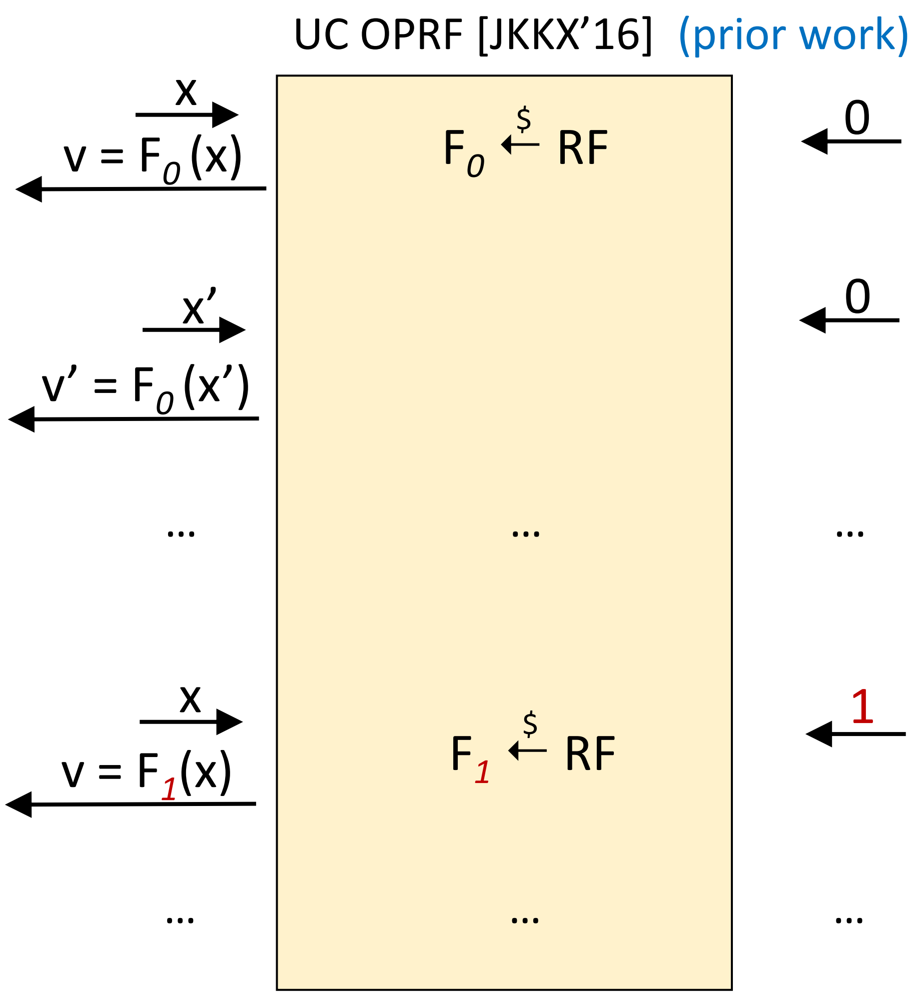
<u>Correlated UC OPRF</u> (this paper)

- $PRF_{k,\delta}$ and $PRF_{k*,\delta*}$ can be correlated on at most <u>one argument x*</u> $\forall(k^*,\delta^*) \neq (k,\delta)$
- On all $x \neq x^*$ these are $\approx$ indep. RF's
- realized by mult-blinded Hashed-DH (under Gap^+ OneMore-DH in ROM)

> DDH^+ oracle, on (A,B,A',B',C) replies 1 iff
>
> C = DH(A,B) · DH(A',B')
>
> [holds in GGM, implied by bilinear DH]

When is *Correlated* OPRF safe to use?

First, compare to the OPRF of [JKKX'16], realized by exp-blinded Hash DH:



Implementation: PRF keys
UC abstraction: pointers to RF's

UC OPRF [JKKX'16] model implies attacker S* actions are ≈ choosing between independent RF's

When is *Correlated* OPRF safe to use? Think Password Auth…

(= UC functionality realized by mult-blinded Hashed DH)

In Password Authentication

x = client's password

UC Correlated OPRF (this paper)

S (attacker)

Initialization: v_0 *will be used for authentication to S*

x →

$v_0 = F_0(x)$ ←

$F_0 \xleftarrow{\$} RF$

← 0

n authentication sessions: application tests if $v_i=v_0$

x →

$v_1 = F_1(x)$ ←

$F_1 \xleftarrow{\$} RF$

s.t. $F_1(x_1) = F_0(x_1)$

← 1, CL ={ (0,x_1) }

x →

$v_2 = F_2(x)$ ←

$F_2 \xleftarrow{\$} RF$

s.t. $F_2(x_2) = F_0(x_2)$

& ….

← 2, CL ={ (0,x_2), (1,…) }

…

…

x →

$v_n = F_3(x)$ ←

$F_n \xleftarrow{\$} RF$

s.t. $F_n(x_n) = F_0(x_n)$

& ….

← n, CL ={ (0,x_n), (1,…), …}

n online password tests: $x_1, x_2, …, x_n$

Correlated OPRF creates avenue for online password tests

only one test per session: attack avenue already part of PAKE

→ OPAQUE aPAKE can use mult-blinded Hashed DH

Conclusions

- Mult-blinded Hashed DH realizes UC Correlated OPRF
  - relaxation of UC OPRF [JKKX'16]
- Correlated OPRF can create on-line password test attacks
  - only one per protocol instance, like in PAKE
  - In threshold OPRF it can create attack avenue for 1 malicious server *[see the paper]*
- It can be used if application already has on-line tests
  - Password-Authentication, e.g. OPAQUE
    - → modified OPAQUE reduces cost of strong asymmetric PAKE to $\leq 2f + 2v$ exps per party
  - Set Intersection
  - other?
  - Warning: If OPRF key is re-used then you must verify (i.e. prove) that Correlated OPRF suffices