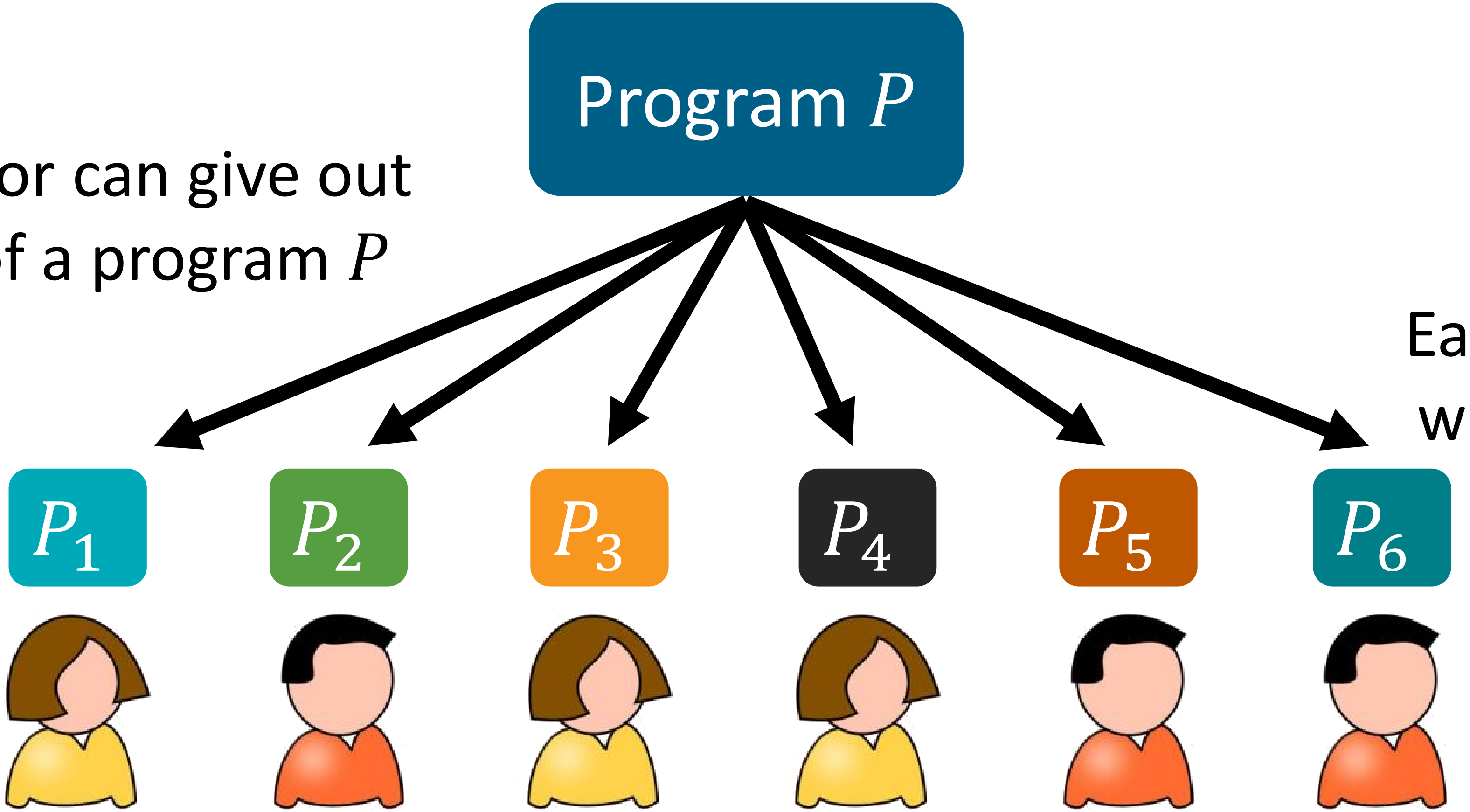
Traceable PRFs: Full Collusion Resistance and Active Security

Sarasij Maitra and David Wu

Traceable Cryptography

Distributor can give out copies of a program P



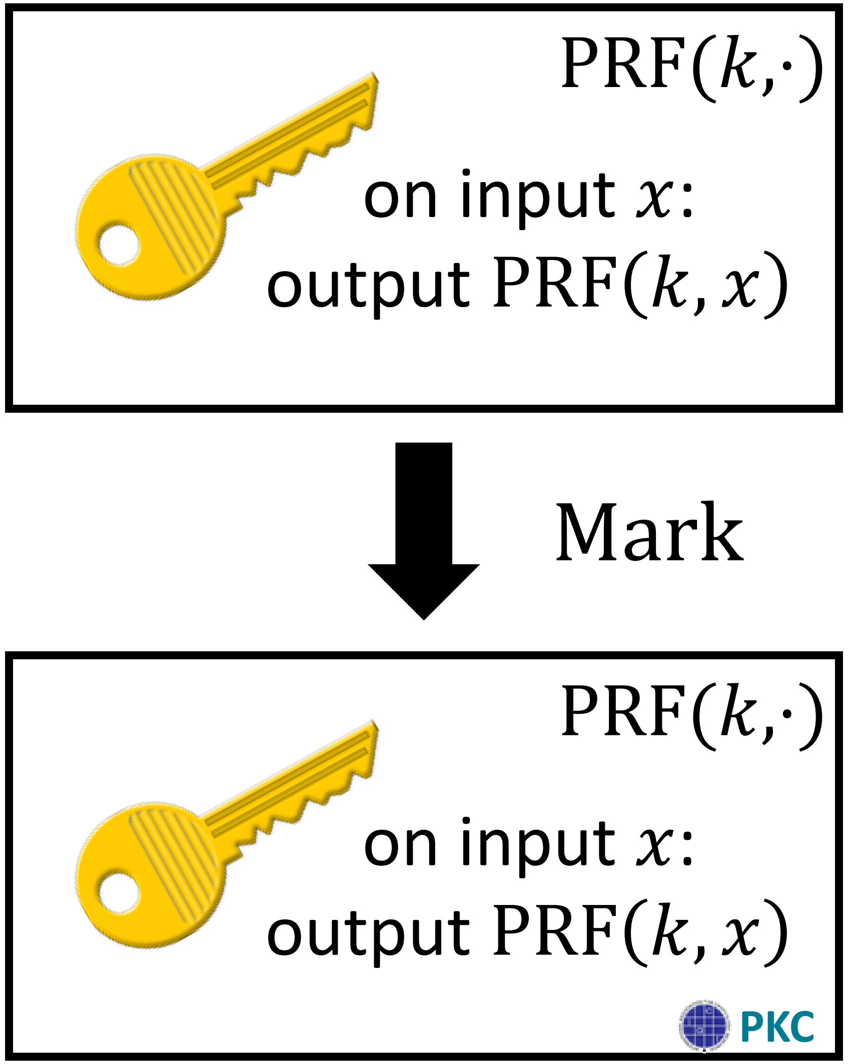
Each copy *marked* with an identifier

Goal: cannot create a new copy that does not contain the identifier

Useful for protecting against unauthorized distribution of software

Traceable PRFs

[GKWW21]



$\mathrm{PRF}(k,\cdot)$

on input x:
output $\mathrm{PRF}(k,x)$

Mark

$\mathrm{PRF}(k,\cdot)$

on input x:
output $\mathrm{PRF}(k,x)$

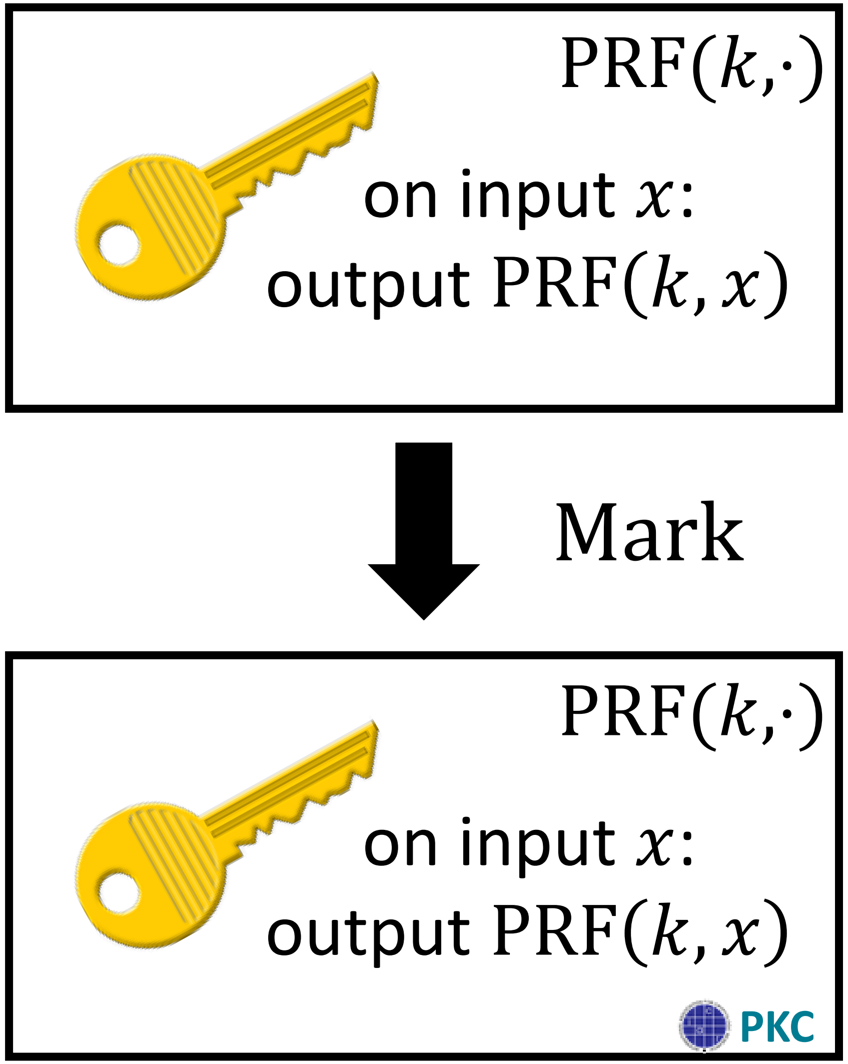
Program implements a pseudorandom function (PRF)

Marking algorithm embeds a *mark* (i.e., an identifier into the program)

Conceptually similar to watermarking, but provides much stronger security guarantees

Traceable PRFs

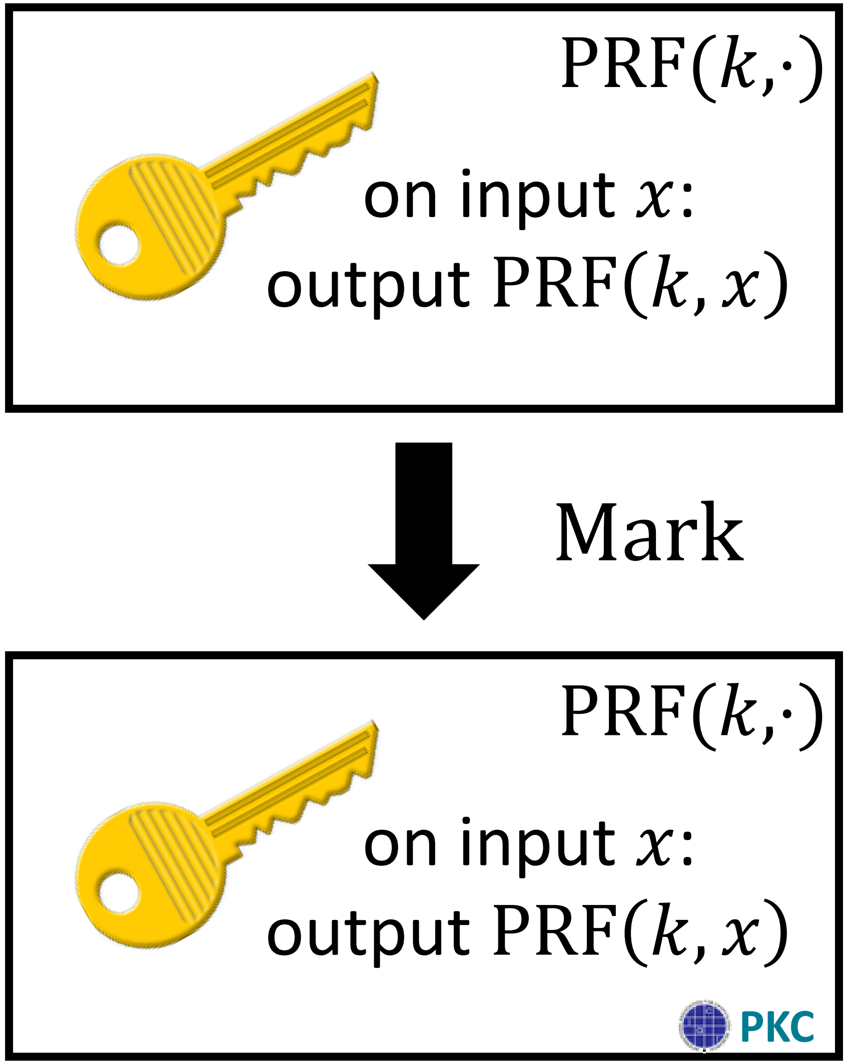
[GKWW21]



Marking security (informal):
if program C can *distinguish*
$\mathrm{PRF}(k,\cdot)$ from random, then mark
should be preserved

Traceable PRFs

[GKWW21]



Primitive suffices for realizing primitives like traitor tracing (since PRF implies encryption)

Marking security (informal):
if program C can *distinguish* $\mathrm{PRF}(k,\cdot)$ from random, then mark should be preserved

Existing Constructions of Traceable PRFs

[GKWW21]

Assuming LWE, there exists a single-key traceable PRF (with secret tracing)

- Security holds only if adversary sees a single marked program
- Completely broken if adversary sees even two marked programs

Assuming indistinguishability obfuscation and injective one-way functions, there exists a fully collusion resistant traceable PRF (with public tracing)

Can we construct collusion-resistant traceable PRFs from LWE?

This Work

A generic approach to upgrade single-key traceable PRF into a fully collusion resistant traceable PRF via fingerprinting codes

Information-theoretic primitive

Corollary. *Assuming LWE, there exists a fully collusion resistant traceable PRF (with secret tracing)*

Caveat: scheme only supports polynomial identity space

Fingerprinting Codes

[BS95, Tar03]

Codewords

1	1	0	0	0	1	1	0	0	0
2	1	0	1	0	1	1	0	1	1
3	1	1	1	1	1	1	1	1	1

Fingerprinting Codes

[BS95, Tar03]

Codewords

1	1	0	0	0	1	1	0	0	0
2	1	0	1	0	1	1	0	1	1
3	1	1	1	1	1	1	1	1	1
	1	?	?	?	1	1	?	?	?

Security: adversary's codeword decodes to one of the codewords it was given

Adversary can craft a codeword where every position is consistent with at least one of the codewords it has

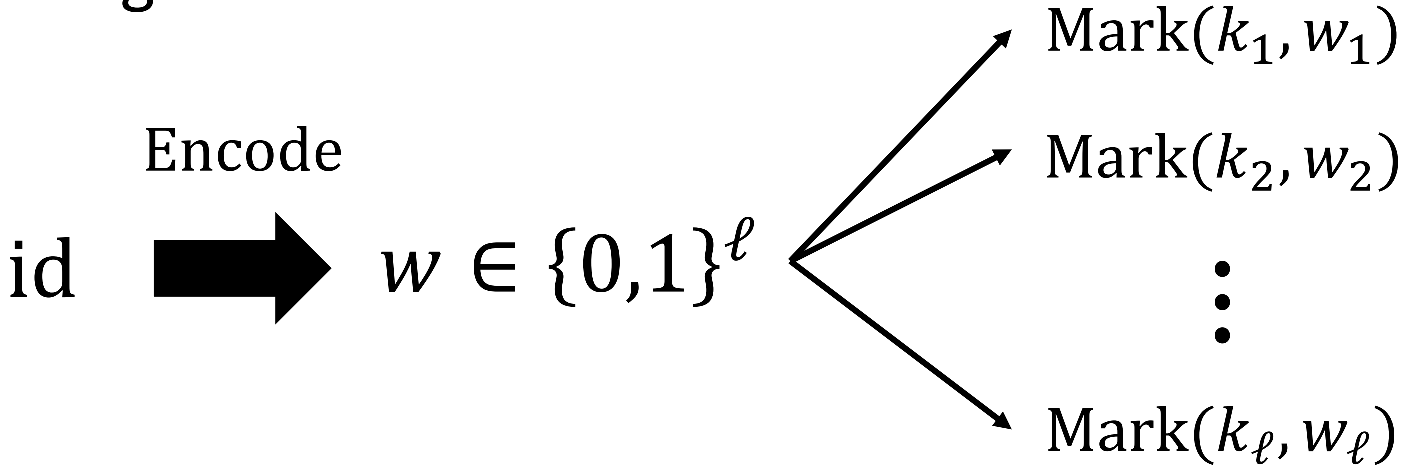
Construction Overview

Let ℓ be the length of the fingerprinting code

Traceable PRF consists of ℓ copies of the single-key traceable PRF:

$$\mathrm{PRF}\big((k_1, \ldots, k_\ell), x\big) = \bigoplus_{i \in [\ell]} \mathrm{PRF}(k_i, x)$$

Marking:



id $\xrightarrow{\text{Encode}}$ $w \in \{0,1\}^\ell$

$\mathrm{Mark}(k_1, w_1)$

$\mathrm{Mark}(k_2, w_2)$

⋮

$\mathrm{Mark}(k_\ell, w_\ell)$

i^{th} key will be marked with i^{th} bit of codeword

Construction Overview

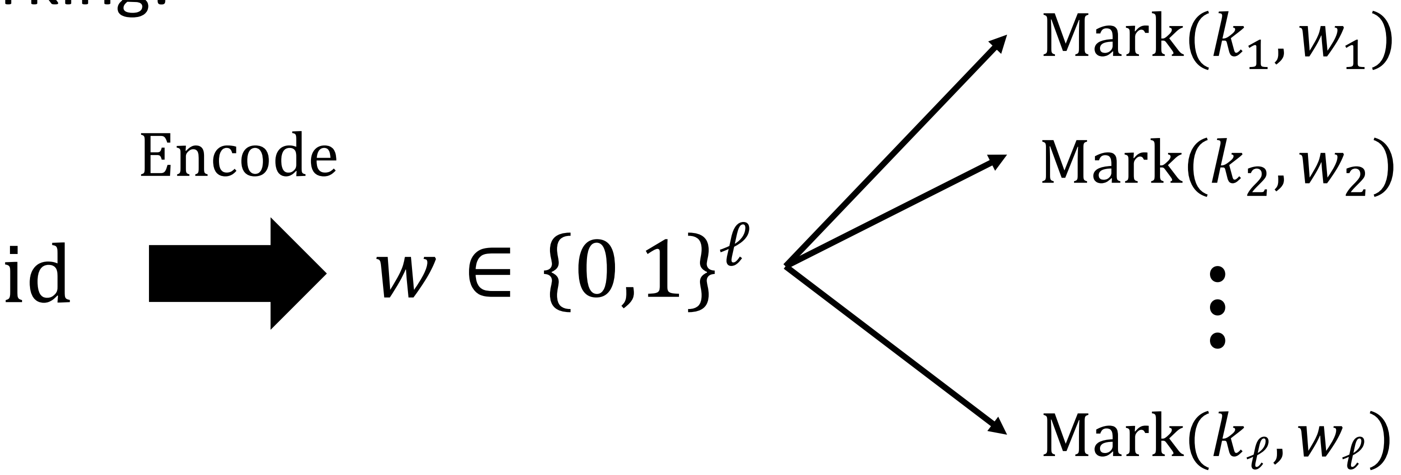
1	1	0	0	0	1	1	0	0	0
3	1	1	1	1	1	1	1	1	1

Single-key security enforces constraint of fingerprinting code model

Observation: For positions where all codewords agree, adversary only sees **1 marked key**

Security reduces to that of fingerprinting code

Marking:



$$\mathrm{id} \xrightarrow{\text{Encode}} w \in \{0,1\}^{\ell} \to \begin{cases} \mathrm{Mark}(k_1, w_1) \\ \mathrm{Mark}(k_2, w_2) \\ \vdots \\ \mathrm{Mark}(k_\ell, w_\ell) \end{cases}$$

i^{th} key will be marked with i^{th} bit of codeword

Summary

A generic approach to upgrade single-key traceable PRF into a fully collusion resistant traceable PRF via fingerprinting codes

Corollary. *Assuming LWE, there exists a fully collusion resistant traceable PRF (with secret tracing)*

Also: approach also useful to achieve *active* security (where adversary has access to tracing oracle)

[see paper for details]

Open Question: collusion resistance for super-polynomial identity space from LWE

Thank you!

https://eprint.iacr.org/2021/1675.pdf