White Box Remote Procedure Call

Work in progress

Amir Herzberg
Haya Shulman
Bar Ilan university

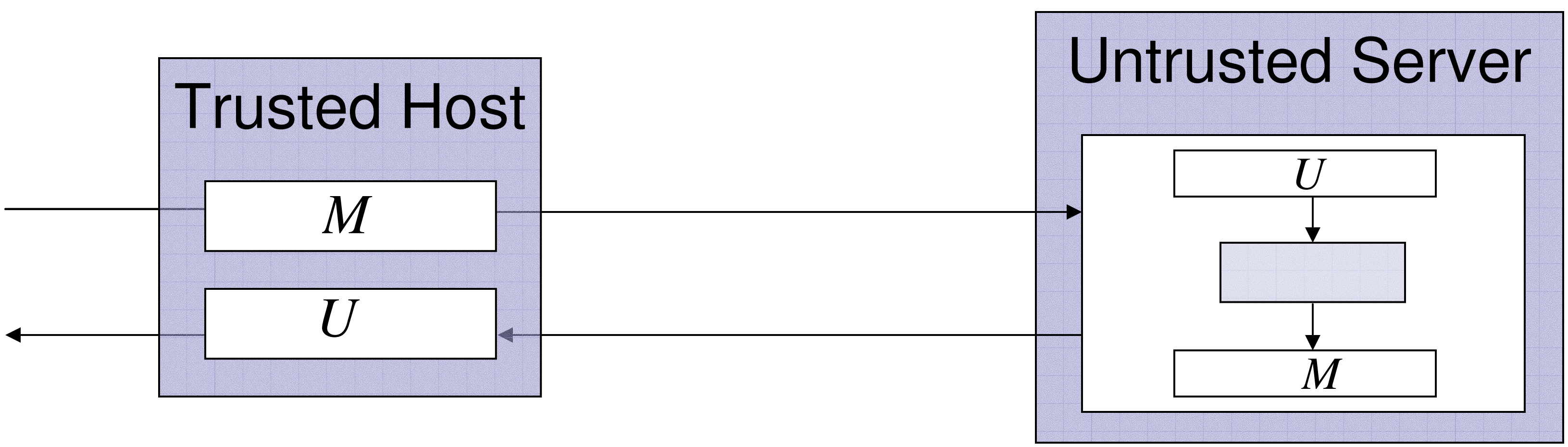
Amitabh Saxena
Bruno Crispo
University of Trento

Talk Outline

- Introducing WBRPC
- Defining Security Specifications
- White Box RPC vs. Obfuscators
- WBRPC Robust Combiner
- Universal White Box RPC
- Conclusions

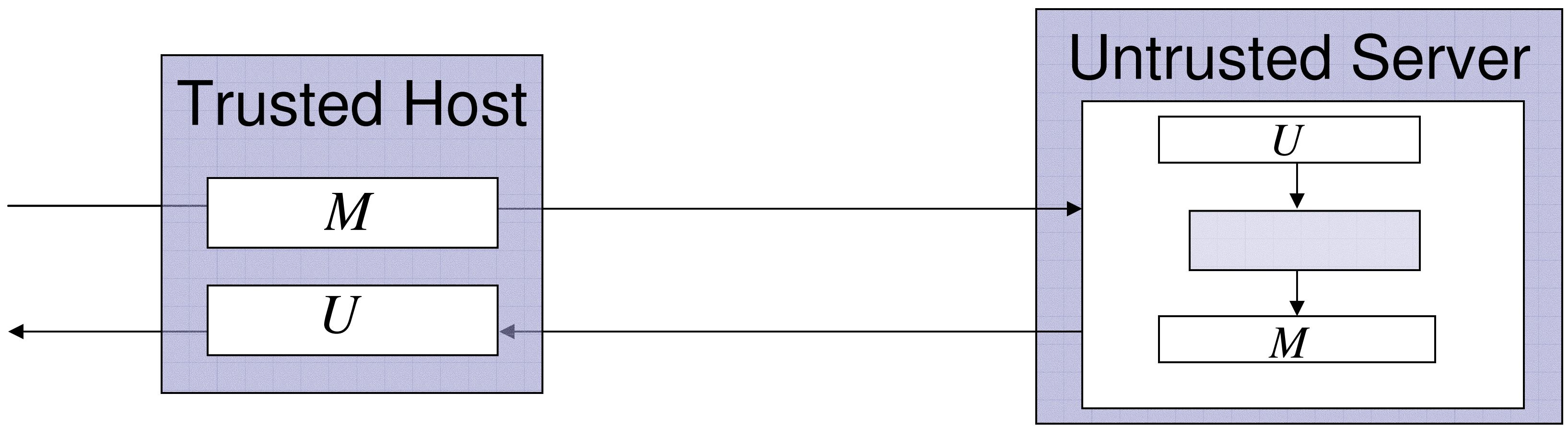
Remote Procedure Call

- Known concept in Software engineering
  - i.e. a popular paradigm for implementing client-server model of distributed computing
  - Allows a program to cause a procedure to execute in another address space
  - Same code irrespective of whether the subroutine is local to the executing program or remote



Remote Procedure Call - *Marshalling*

- Packing of function parameters into a message packet
  - Marshal or unmarshal the parameters of an RPC
    - Client marshals the arguments into a message
    - Server unmarshals the arguments and uses them to invoke the service function
  - On return
    - Server marshals return values
    - Client unmarshals return values, and returns to the client program



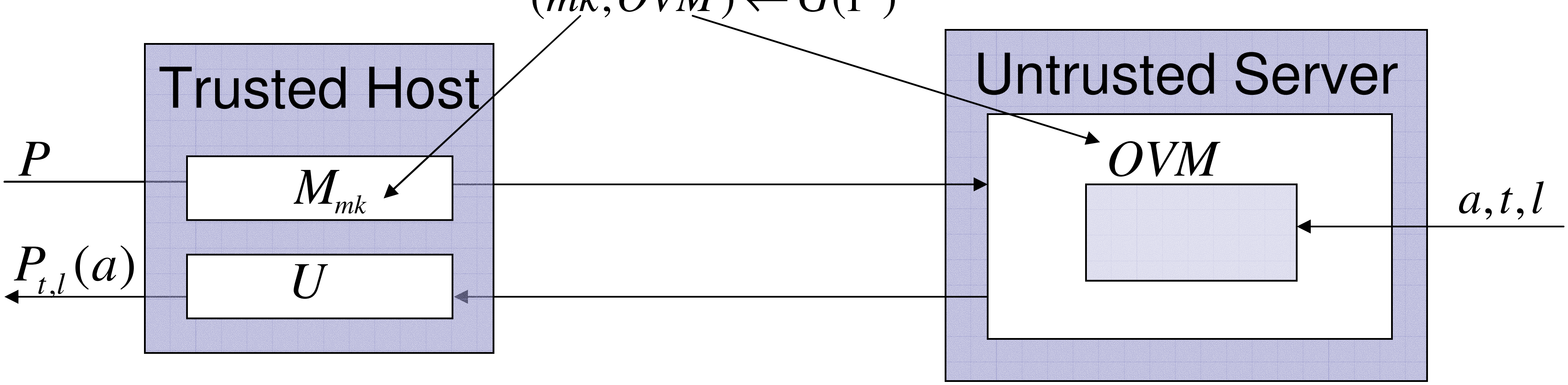
Goals and Motivation

- Software only security
- Execute programs on an untrusted host in a secure manner
- Protection for arbitrary function
  - As opposed to specific function (e.g. WB-AES, WB-DES)
- Provide Integrity and Confidentiality
  - Protect program and data
- Efficiency
- Well defined game based specifications
- Tolerant design (Robust Combiners)

White Box Remote Procedure Call

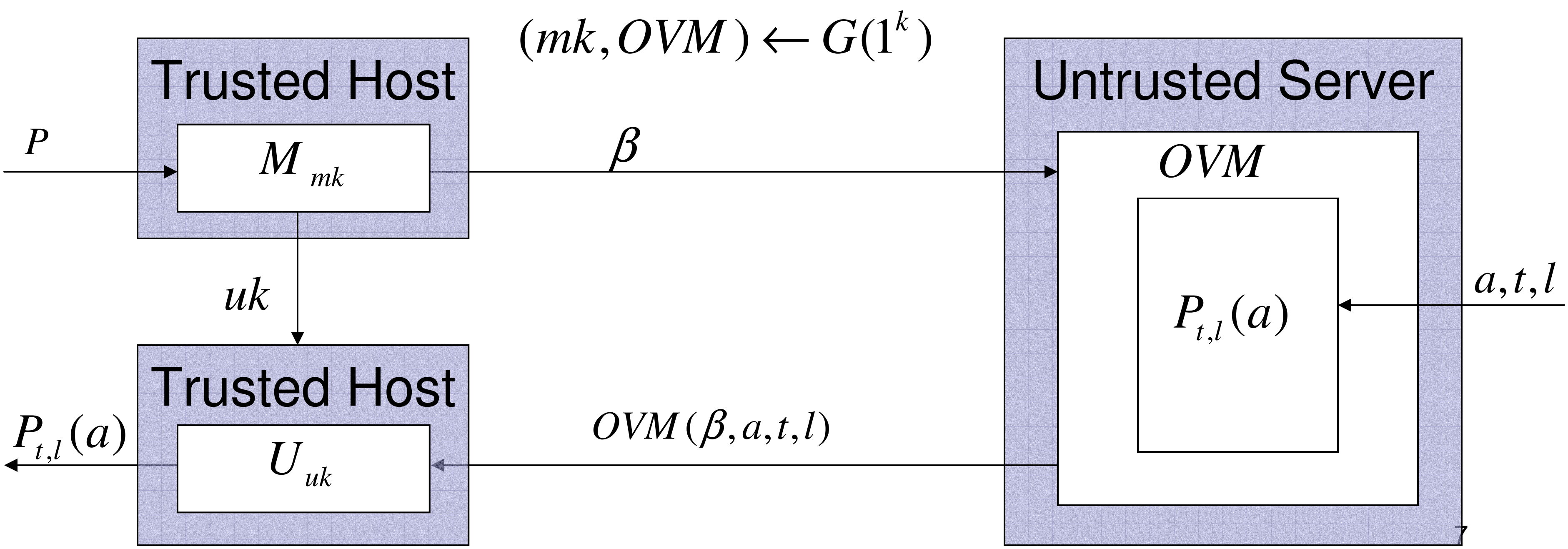
- *Code execution in an untrusted remote environment*
  - ❑ Mobile Agents
  - ❑ Grid Computing
  - ❑ Electronic Voting
  - ❑ Queries on private DB
  - ❑ Multiplayer Games
  - ❑ Unselfish cooperation (VoIP, P2P nets)
  - ❑ Intellectual Property Rights
  - ❑ eCash
- *l bits (padded/ truncated) output of P(a) after running for t steps*
- *A WBRPC scheme* W *is a tuple of PPT algorithms* (G, M, U)

$$(mk, OVM) \leftarrow G(1^k)$$



White Box Remote Procedure Call

- *A WBRPC is a tuple of PPT algorithms <G,M,U>*
  - $\forall (mk, OVM) \in G(1^k), a \in \{0,1\}^*, P \in TM, t \in 1^*, l \in \mathrm{N}$ holds
  - $(uk, \beta) \leftarrow M_{mk}(P)$, *where* $|uk| \le k^{\alpha}$, *s.t.* $\exists \alpha \in N$
  - $P_{t,l}(a) \leftarrow U_{uk}(OVM(\beta, a, t, l))$

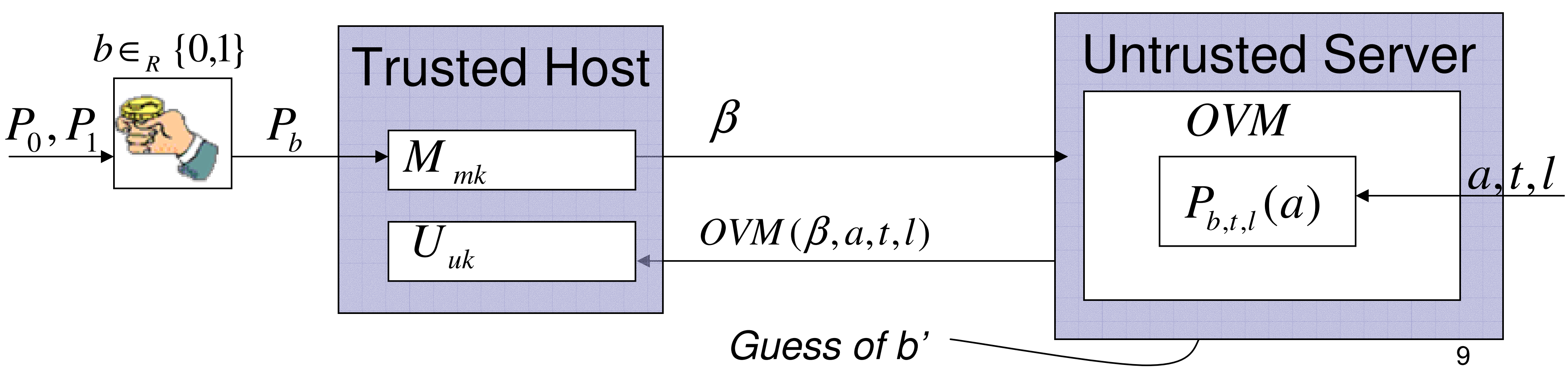


WBRPC Security Requirements

- Protect the user of the trusted host
  - Privacy of *P* (e.g. the secret key in *P*)
  - Unforgeability
- Protect the privacy of auxiliary input *a (i.e. untrusted host)*
  - The adversary cannot learn anything new about *a*
  - Validity (e.g. execute only valid programs)

WBRPC Privacy (IND) Specification

- Protect the trusted host (from malicious server)
- Requirement
  - Indistinguishability of the input programs
  - For example, hide a key or data inside P



WBRPC IND Experiment $Expt^{IND}_{W,A_1,A_2}(k)$

$(mk, OVM) \leftarrow G(1^k)$

$(P_0, P_1, s) \leftarrow A_1^{MO(\cdot)}(OVM)$

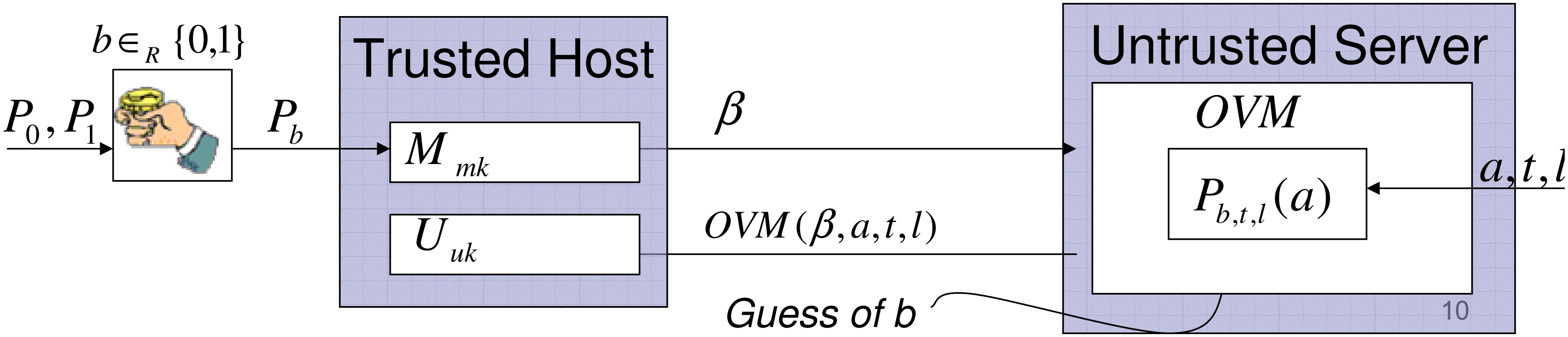
$b \in_R \{0,1\}$

$(uk, \beta) \leftarrow M_{mk}(P_b)$

$b' \leftarrow A_2^{MO(\cdot)}(s)$

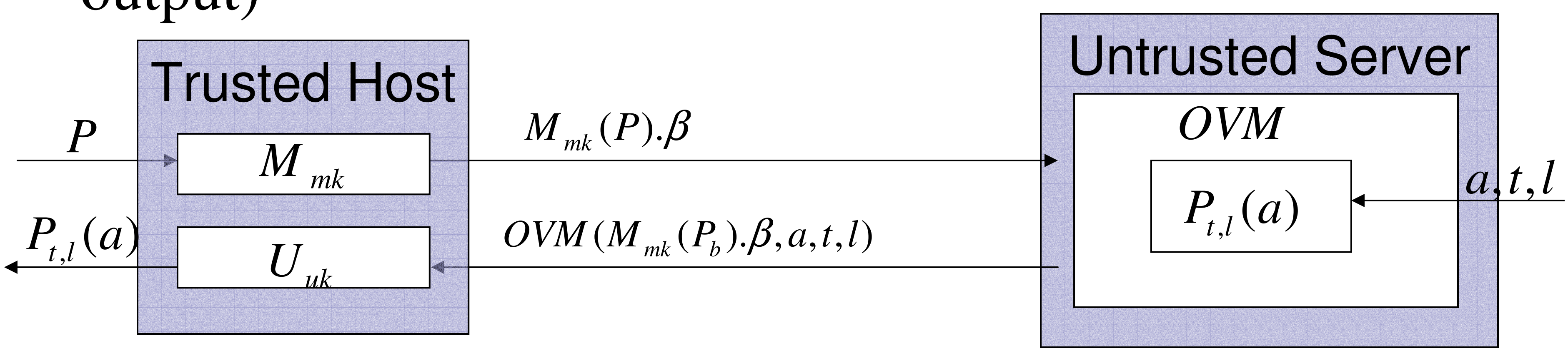
$if\ b = b'\ and\ |P_0| = |P_1|\ return\ 1,\ else\ return\ 0$

```
MO(P) /* marshaling oracle */
{
        return M_mk(P)
}
```



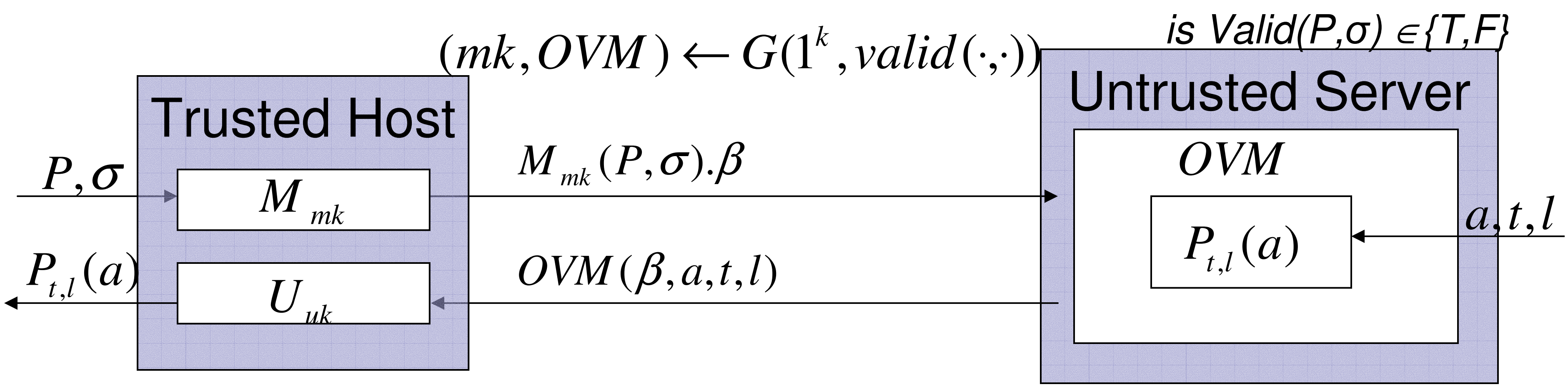
WBRPC Unforgeability Specification

- Protect the trusted host (from malicious server)
- Requirement
  - Detect output forgery, i.e. output which is not $P_{t,l}(a)$ for any a
- The *mk* key can be either public or private
- The *uk* key can be public (to only authenticate non-secret output)



WBRPC Privacy Specification (Intuition)

- Protect the owner of the untrusted server
- Requirement
  - Protect confidentiality of auxiliary inputs
  - Expose only the output, i.e. $P_{t,l}(a)$
  - Validate program P by *valid(P, σ), s.t.* σ is a validation parameter received along with P



WBRPC Computational Complexity

- Communication Complexity

$$|M_{mk}(P,\sigma).\beta| \le \text{poly}(|P,\sigma|)$$

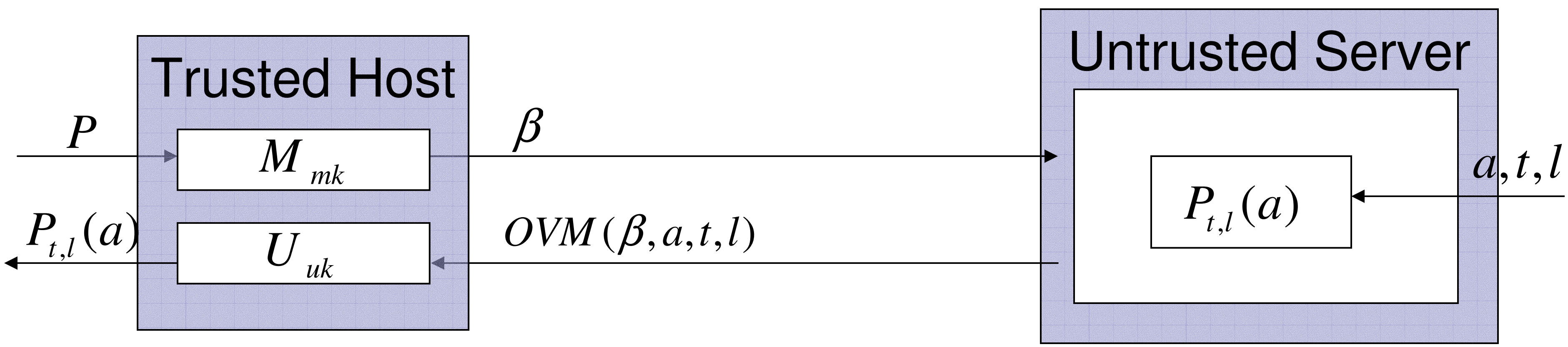
$$|OVM(M_{mk}(P,\sigma).\beta,a,t,l)| \le poly(|P_{t,l}(a)|)$$

- Time Complexity

$$time(M_{mk}(P,\sigma)) \le poly(|P,\sigma|)$$

$$time(U_{uk}(OVM(\beta,a,t,l))) \le poly(P_{t,l}(a))$$

$$time(OVM) \le poly(t)$$



Related Work – Application Specific WBRPC

- Private Searching on Streaming data
  - Rafail Ostrovsky and William E.Skeith
  - Concept similar to WBRPC
    - However for specific task only
    - Theoretical result (inefficient constructions)
  - WBRPC reducible to their definitions (the opposite is not true)
  - Achieves some of the security specifications of WBRPC

Talk Outline

- Introducing WBRPC
- Defining Security Specifications
- White Box RPC vs. Obfuscators
- WBRPC Robust Combiner
- Universal White Box RPC
- Conclusions

Obfuscator "Definition"

- Semantics-preserving transform of code that renders it more secure against confidentiality attacks
- "Formally"
  - *O(P)* computes the same function as *P*
  - *O(P)* time (resp. size) complexity is polynomial in *P*'s
  - [Barak et al.] Virtual black box
    - An obfuscated program reveals no more information than a black box access to it

Other Security Variants

- [Barak et al] TM indistinguishability
- Best Possible Obfuscation
  - Shafi Goldwasser and Guy Rothblum
- Simulation based definition (the functions to be obfuscated are chosen at random)
  - Dennis Hofheinz, John Malone-Lee, Martijn Stam
- Securily Obfuscating Re-Encryption
  - Susan Hohenberger, Guy Rothblum, Abhi Shelat, Vinod Vaikuntanathan

Obfuscation Impossibility Results – [Barak et al]

- There does not exist a general obfuscator for arbitrary function families
- There exist non-obfuscatable functions
  - e.g. contrived encryption/ signature/ MAC schemes

WBRPC vs. Obfuscators

	WBRPC	Obfuscators
•IND of programs	•Yes	•Output known (only if BB IND)
•UNF of output	•Yes	•No
•Availability	•For some applications yes (e.g. Ostrovsky) •Universal WBRPC ⇒ WBRPC for every TM •Proposals WB-DES, WB-AES	•No, for all TMs •Yes, for Point Functions (e.g. Access control) Re-Encryption

WBRPC vs. Obfuscators

	WBRPC	Obfuscators
Hiding auxiliary input	Yes (provides privacy of inputs and validity of programs)	No
Untrusted Host Outputs	Encrypted output	Output of the original function

Talk Outline

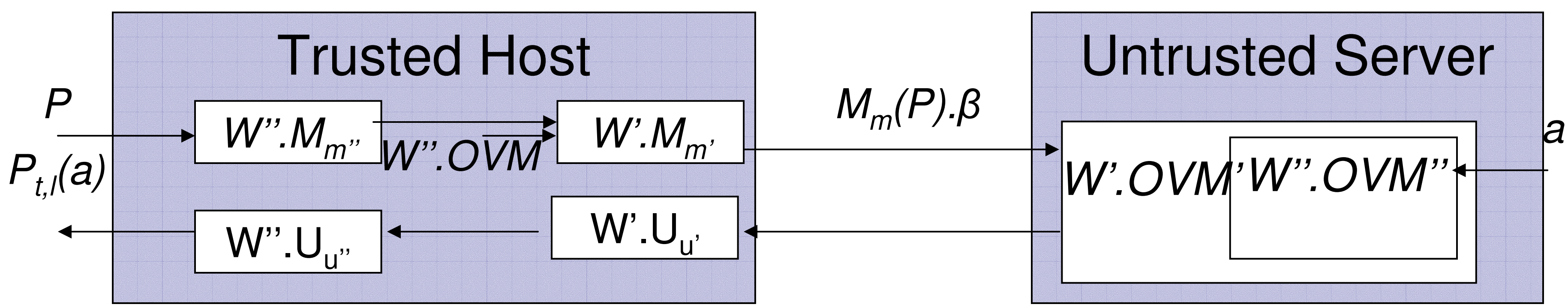
- Introducing WBRPC
- White Box RPC vs. Obfuscators
- Defining Security Specifications
- WBRPC Robust Combiner
- Universal White Box RPC
- Conclusions

White Box RPC Robust Combiner

- Given *two* candidate White-Box RPCs W', W''
- Can we *combine* them into one White-Box RPC
  - *W* ← *W' • W''*
  - s.t. *W* is a secure white box RPC provided one of *W', W''* is secure
  - A robust combiner
- [H05] : Robust Combiners, Definitions, Constructions (e.g. encryption, commitment schemes)
  - Also other works…

White Box RPC Robust Combiner

- Given *two* candidate White-Box RPCs *W'*, *W''*
- Idea: run *W''* under *W'!*



White Box RPC Robust Combiner

Generation Procedure $\mathcal{G}(1^k)$

$\langle mk', OVM' \rangle \leftarrow \mathcal{G}'(1^k)$

$\langle mk'', OVM'' \rangle \leftarrow \mathcal{G}''(1^k)$

$mk = \langle mk', mk'', OVM' \rangle$

$OVM = OVM''$

$return\ \langle mk, OVM \rangle$

Marshalling $\mathcal{M}_{\langle mk',\ mk'' \rangle}(P) = \langle uk, \mathcal{M}''_{mk''}(P') \rangle\ s.t.$

$uk = \langle uk', uk'' \rangle$

$\langle uk', \beta' \rangle \leftarrow \mathcal{M}'_{mk'}(P)$

$\langle uk'', \beta'' \rangle \leftarrow \mathcal{M}''_{mk''}(P')$

Program $P'\{$

`Read` a `of the input tape`

`return` $[$`OVM'`$(\mathcal{M}'_{mk'}(P), a)];$

$\}$

Unmarshalling $\mathcal{U}_{\langle uk',\ uk'' \rangle}(\omega) = \mathcal{U}'_{uk'}(\mathcal{U}''_{uk''}(\omega))$

WBRPC Combiner, Theorem

- Theorem
  - $W \leftarrow W' \bullet W''$ is Robust for indistinguishability

- *Proof*, *c*onsider the following lemmas
  - Lemma 1
    - Given W' is indistinguishable, $W = W' \bullet W''$ is indistinguishable
    - Let W' be an IND-secure, then given a *PPT* Adversary $A=(A_1,A_2)$, there exists a *PPT* Adversary $A'=(A'_1,A'_2)$ s.t. for infinitely many k's

$$Adv^{IND}_{W',A',\varphi}(k) = Adv^{IND}_{W,A,\varphi}(k)$$

  - Lemma 2
    - Identical for W''

Proof of Lemma 1: W' IND => $W \leftarrow W' \bullet W''$ IND

- Given a *PPT* $A=(A_1,A_2)$ construct a *PPT* $A'=(A'_1,A'_2)$ against W' that has black box access to A and W' (i.e. marshalling oracle and *OVM)*
- Consider following programs for A' algorithm and $MO(\cdot)$ oracle

- A' operates according to the steps defined in the indistinguishability experiment

Adversary A' against W'

Algorithm $A_1'^{\mathcal{MO}'(\cdot)}(\text{OVM'})$

1. $\langle mk'', \text{OVM''} \rangle \leftarrow \mathcal{G}''(1^k)$
2. $\langle P_0, P_1, s \rangle \leftarrow A_1^{\mathcal{MP}(\cdot)}(\text{OVM''})$
3. Return $(\langle P_0, P_1 \rangle, \langle mk'', \text{OVM''}, s \rangle)$

Algorithm $A_2'^{\mathcal{MO}'(\cdot)}(\text{OVM'}, \mathcal{M}'_{mk'}(P_b), \langle mk'', \text{OVM''}, s \rangle)$

1. Let P' be the following program $P_b'\{$

   `Read` a `of the input tape`

   `return` $[$`OVM'`$(\mathcal{MO}'(P_b), a)]$;

   $\}$
2. $\omega^* = \mathcal{M}''_{mk''}(P_b')$
3. $b' \leftarrow A_2^{\mathcal{MP}(\cdot)}(\text{OVM''}, \omega^*, s)$
4. Output b'

Marshaling procedure accessed by A using MO(·) oracle

Marshalling Procedure $\mathcal{MP}(P)$

1. Let P' be the following program, $P'\{$

   Read a ***of the input tape***

   `return` $[$`OVM'`$(\mathcal{MO}'(P), a)];$

   $\}$

   return (P', mk'')

2. $else$

   return $\mathcal{M}''_{mk''}(P')$

White Box RPC Robust Combiner

- The success advantage of *A* in the IND experiment is equivalent to the success advantage of *A* in the IND simulation executed by *A'*
- *Claim*
  - Let ***r*** denote a sequence of random coins used in a specific execution of IND experiment and let $Expt^{IND}_{W,A,\varphi}(k;r)$
  - By the design of the experiment, the algorithm *A'* and by the implementation of the marshaling oracle *MO'(·)* it follows

$$Expt^{IND}_{W',A',\varphi}(k;r) = Expt^{IND}_{W,A,\varphi}(k;r)$$

Talk Outline

- Introducing WBRPC
- White Box RPC vs. Obfuscators
- Defining Security Specifications
- WBRPC Robust Combiner
- Universal White Box RPC
- Conclusions

Universal White Box RPC

- [Barak et al]
  - No obfuscator for all TM
- Question
  - WBRPC for all TMs?
- Idea, find Universal WBRPC s.t. given
  - Obfuscator for Universal WBRPC we obtain WBRPC for all TMs
  - WBRPC for Universal WBRPC obtain WBRPC for all TMs

Universal WBRPC using Obfuscator

Program $\mathcal{G}\ (1^k)$ {

  $(e', d') \xleftarrow{R} \mathcal{G}'_E(1^k)$

  String $VM :=$ "Program VM $(C, a, t, l; r)$ {

    $d' :=$ " $\|\, d' \,\|$ " /* hardcode d' */

    $(P, e'', s, r'') \leftarrow$ Decrypt(d', C)

    $c \leftarrow$ Encrypt$(e'', \langle$UTM$(P, t, l, a)\rangle, r'')$

    $\sigma \leftarrow$ Sign(s, c)

    Return $\langle c, \sigma \rangle$

    }"

  String FN;

  $OVM \leftarrow O($Compile$(FN \| VM))$

  Return (e', OVM)

}

Universal WBRPC using Obfuscator

String $FN :=$ "

Function Decrypt (d', C) { /* Returns $\mathcal{D}'_{d'}(C)$ */ }

Function UTM (P, t, l, a) {/* Returns $P_{t,l}(a)$ */ }

Function Encrypt (e'', m, r'') {/* Returns $\mathcal{E}''_{e''}(m; r'')$ */

Function Sign (s, m) {/* Returns $\mathcal{S}_s(m)$ */ }"

Universal WBRPC using Obfuscator

$\texttt{Program}\ \mathcal{M}\ (e', P, t, l)\ \{$

$\quad (e'', d'') \xleftarrow{R} \mathcal{G}''_E(1^k)$

$\quad (s, v) \xleftarrow{R} \mathcal{G}_S(1^k)$

$\quad r', r'' \xleftarrow{R} \{0, 1\}^k$

$\quad C \leftarrow \mathcal{E}'_{e'}(P, e'', s, r''; r')$

$\quad \texttt{Return}\ (\langle d'', v \rangle, C)$

$\}$

Universal WBRPC using Obfuscator

Program $\mathcal{U}\left(\langle d'', v\rangle, \langle c, \sigma\rangle\right)$ {
    If $(\mathcal{V}_v(c, \sigma) = \mathsf{False})$
     then Return $\perp$
     else {
        $(y, ar) \leftarrow \mathcal{D}''_{d''}(c)$
        Return y
       }
}

Universal White Box RPC

- Theorem
  - Construct a WBRPC (resp. Obfuscator) for specific given program, obtain a Universal WBRPC for every program
    - Implication: Universal WBRPC $\Rightarrow$ WBRPC for all TMs

Open problems

- WBRPC is an alternative model for SW ‘hardening’
- We believe it is reasonable to assume that all programs can be White Boxed
- Provably-secure WBRPC scheme
- design a practical WBRPC scheme
- Presented Robust Combiner for WBRPC
  - Secure if at least one of the underlying candidates is secure
- This motivates exploring other, related, weaker or stronger notions of white-box security
  - Namely, to try to find some notion that we can prove realizable or unrealizable

Questions

- Fin.