# Robust Combiner for White Box Security

**Amir Herzberg**

**Haya Shulman**

Bar Ilan university

# Robust Combiner for White Box Security: **Outline**

- White-box security
  - Definition, applications, negative results
- WBRPE: Definition and properties
- Using Cryptanalysis-Proven Schemes and Robust Combiners
- WBRPE Robust Combiner

# White Box Security

- White-box security:
  - Program running in hostile environment
  - May contain proprietary secrets (e.g. keys)
  - Ensure confidentiality of secrets and integrity of execution
- Why is White-Box Security Interesting?
  - Practical applications
    - DRM, Trusted Computing
    - Agents running in (untrusted) marketplace
    - Grid computing… and more
  - No existing practical, secure schemes
    - White-box encryption ?
    - Obfuscators ?
  - Theoretical interest: is white-box security feasible?
    - Negative results: obfuscators

# White-Box Security: Obfuscation

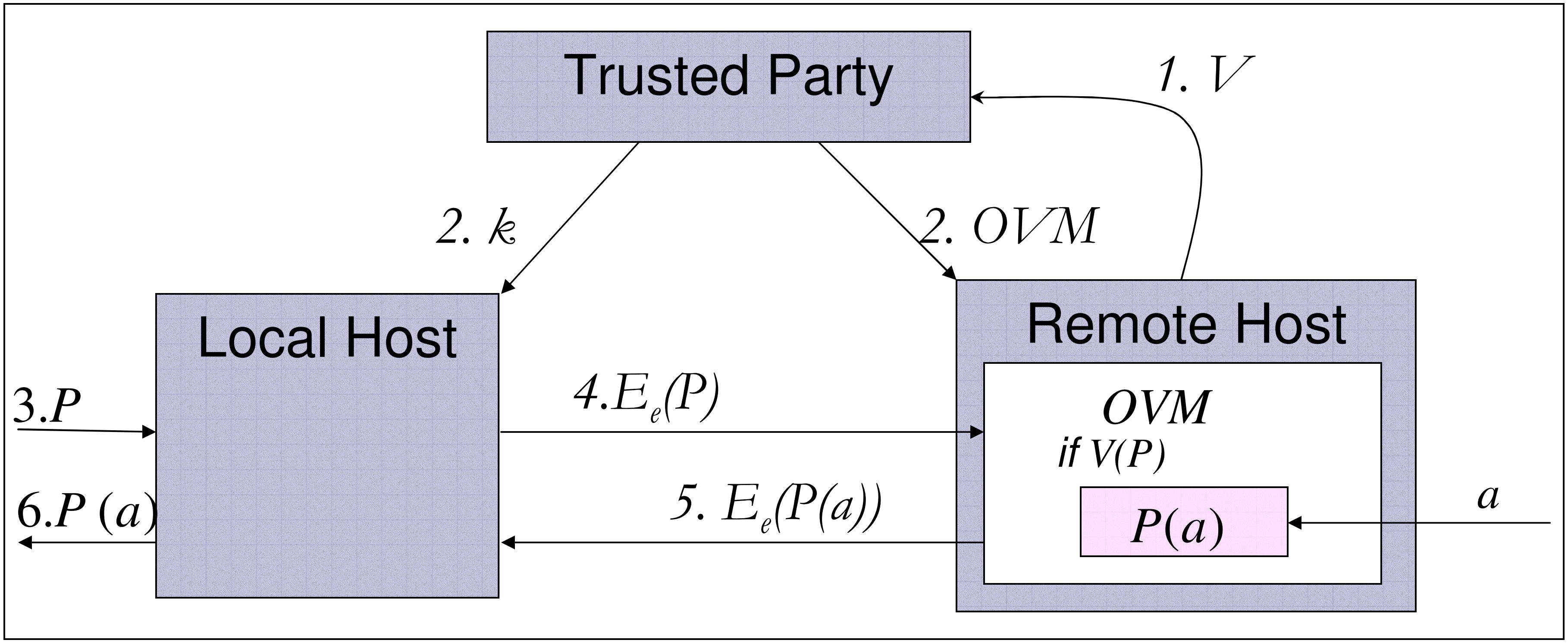
- Most common approach, building block: obfuscators
- Obfuscator O: transforms program P to O(P) s.t.
  - □ O(P) computes same function as P
  - □ Adversary cannot learn more from O(P) than from oracle access to P
- [Barak et al.]: no `obfuscator` for all programs
- [Collberg]: constructions and tools
  - □ But: no secure obfuscator (yet?) to simple modular programs
  - □ At least, no open/published design
- Goal: explore other white-box security primitives
  - □ Avoid impossibility results
  - □ Try to achieve secure, open, practical solutions
  - □ Candidate: WBRPE (White Box Remote Program Execution)

# WBRPE (White-Box Remote Program Execution)

- Program sent by (trusted) local host
- Executed on (potentially hostile) remote host
- Using keys, `OVM` (Obfuscated Virtual Machine) generated by Trusted Third Party
- Security properties:
  - Confidentiality of program sent by local host
  - Confidentiality of the input *a* of remote host
    - By allowing only programs *P* passing validation function *V* (set by remote)
  - Output integrity: output is result of running *P* (over some *a*)
- Efficiency
  - Local host has limited amount of work
  - One-round computation

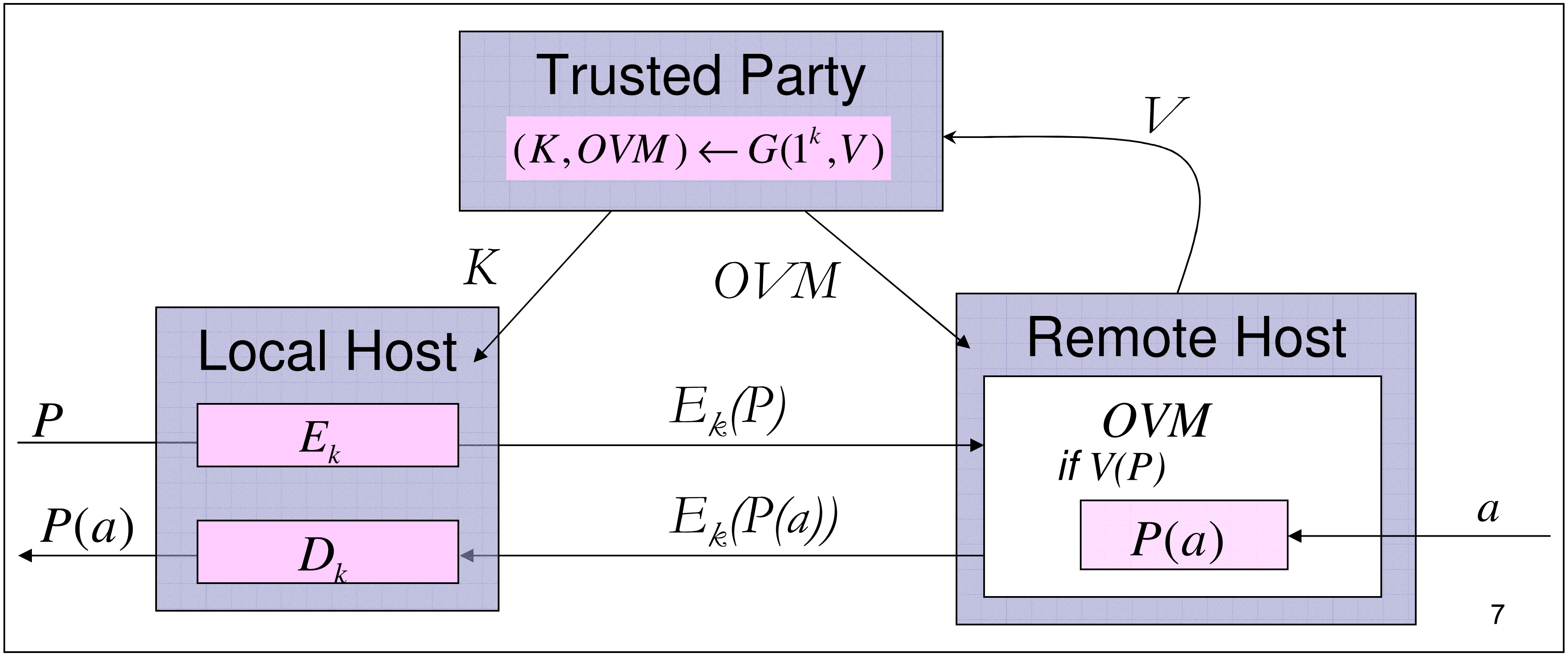
# WBRPE: Entities, Flows

- WBRPE: possible white box security building block
- Entities: Trusted Party, Local Host, Remote Host



# WBRPE: Components (Algorithms)

- Generator $G$ : run by Trusted Party
  - □ Generates key $k$ (for local host)
  - □ And Obfuscated Virtual Machine $OVM$ (for remote host)
- `Encryption' (of program sent by local host)
- `Decryption' (of result sent by remote host)



# WBRPE: Goals and Results

- Reach comparable situation to cryptography:
- Provably secure WBRPE schemes
  - May not be practical (cf. [GM84, OTP])
- Practical, efficient, cryptanalysis-proven WBRPE schemes
  - Secure by evidence of failed cryptanalysis, safety margins
- Results
  - Theoretical feasibility results (provably secure schemes) [next]
  - Robust combiner: given two candidate WBRPE schemes, build one that is secure – if one of the two candidate schemes is secure
    - Allows safety-margins in design

# Robust Combiner for White Box Security: **Outline**

- White-box security
  - □ Definition, applications, negative results
- WBRPE: Definition and properties
- Using Cryptanalysis-Proven Schemes and Robust Combiners
- WBRPE Robust Combiner

# Using Cryptanalysis-Proven Schemes

- We will show that provably-secure WBRPE schemes exist
- Yet, we may use `cryptanalysis-proven` schemes:
  - `Proven` only by failure to break (cryptanalyze)
  - To avoid limitations, e.g. constant # of runs
  - For better (reasonable) efficiency
- Just like for encryption schemes
  - Provably-secure schemes exist ([GM84,…])
  - Yet, we use cryptanalysis-proven schemes: AES, RSA…
- We won't present candidate WBRPE schemes <u>today</u>
- But: we present <u>robust combiner</u> for WBRPE schemes

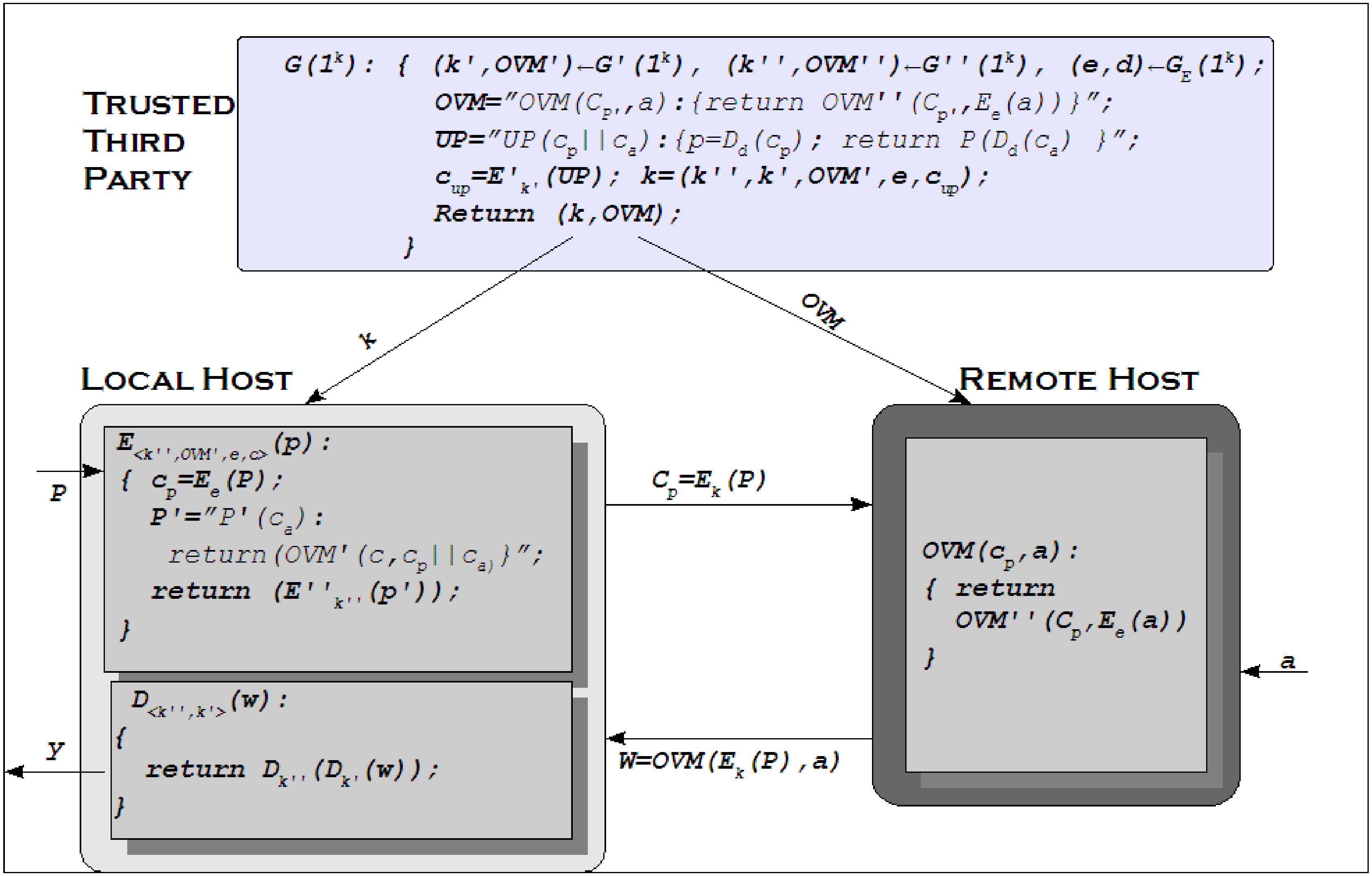
# Robust Combiners: Security by Redundancy

- \`Resilient\` security using multiple schemes:
  - Combine $n$ schemes, $E=C(E_1, E_2,\ldots E_n)$
  - $C$ is a $(t,n)$-robust combiner if:<br>($t$ or more of $E_1,\ldots E_n$ are secure) ➔ $E$ is secure
  - \`Belt and suspenders\` use of cryptanalysis-proven schemes
- Known robust combiners
  - Encryption, Mac/Sign, Commitment,… [H05/8]
    - E.g., cascade encryption: $E_{k1,\ldots kn}(m)=E_{1,k1}(E_{2,k2}(\ldots(E_{n,kn}(m)\ldots))$
  - Oblivious Transfer, PIR, hash, … [HKNRR05,…]
- Our result: robust combiner for WBRPE schemes
  - Esp. important, considering no existing candidates!!

# White Box RPE Robust Combiner

- No established practical WB-security scheme
  - □ So we should robustly-combine candidates!
- Given *two* candidate White-Box RPEs *W'*, *W''*
- Let *W* ← *W'* • *W''* be the cascade of *W'*, *W''*
  - □ As defined in next foil…
- WBRPE Cascade is a robust combiner
  - □ *W* is secure WBRPE if at least <u>one</u> of *W'*, *W''* is secure
  - □ For all WBRPE security specifications

# WBRPE Cascade: a Robust Combiner



TRUSTED THIRD PARTY

```
$G(1^k)$: { $(k', OVM') \leftarrow G'(1^k)$, $(k'', OVM'') \leftarrow G''(1^k)$, $(e,d) \leftarrow G_E(1^k)$;
          OVM="$OVM(C_{p'}, a)$:{return $OVM''(C_{p'}, E_e(a))$}";
          $\overline{UP}$="$UP(c_p || c_a)$:{$p=D_d(c_p)$; return $P(D_d(c_a)$ }";
          $c_{up}=E'_{k'}(\overline{UP})$; $k=(k'', k', OVM', e, c_{up})$;
          Return (k,OVM);
      }
```

LOCAL HOST

```
$E_{<k'', OVM', e, c>}(p)$:
{ $c_p=E_e(P)$;
   $P'$="$P'(c_a)$:
     return(OVM'$(c, c_p || c_{a)}$}";
   return $(E''_{k''}(p'))$;
}
```

```
 $D_{<k'', k'>}(w)$:
{
  return $D_{k''}(D_{k'}(w))$;
}
```

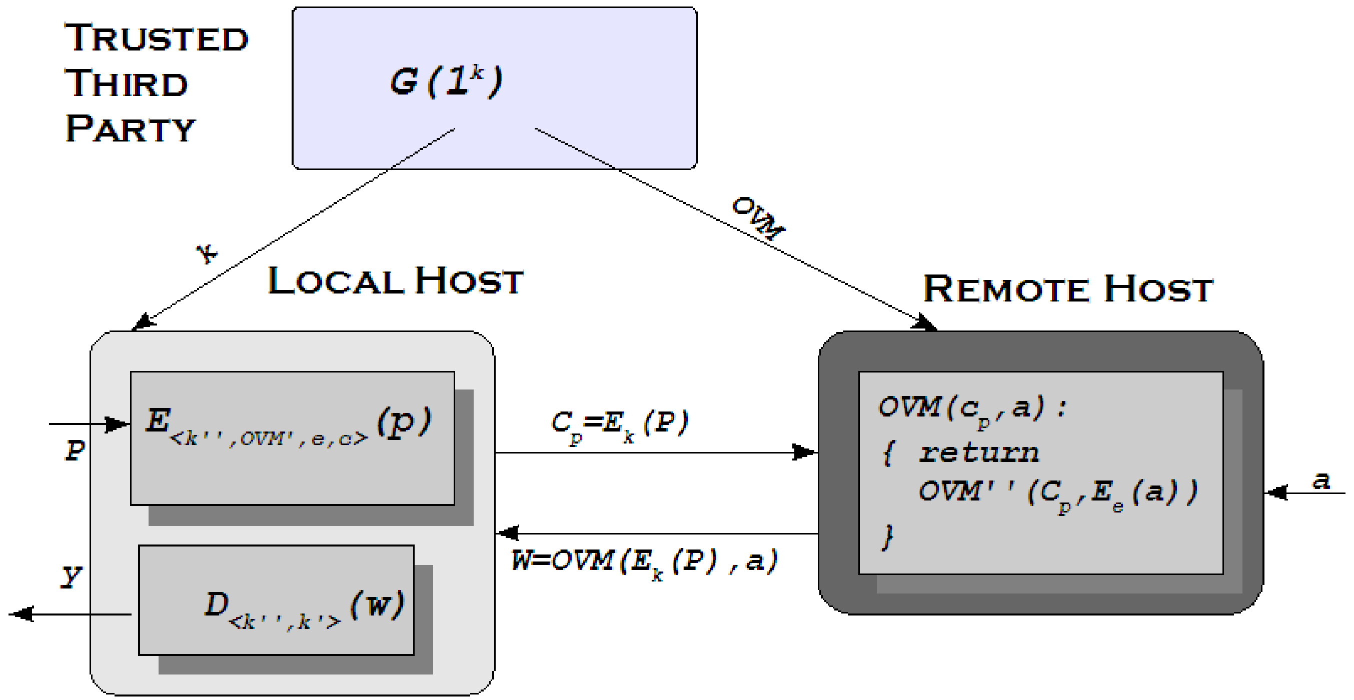
$C_p=E_k(P)$

$W=OVM(E_k(P), a)$

REMOTE HOST

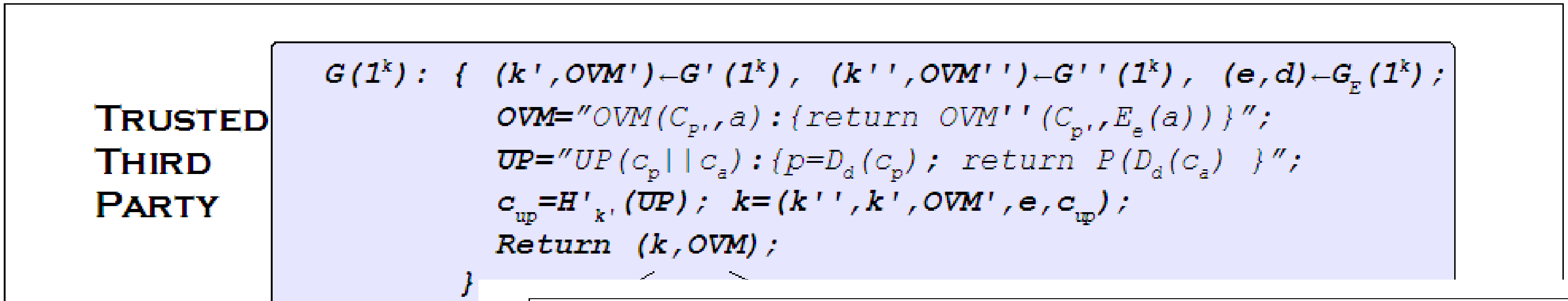
```
$OVM(c_p, a)$:
{ return
  $OVM''(C_p, E_e(a))$
}
```

# WBRPE Cascade: a Robust Combiner (Simplified)



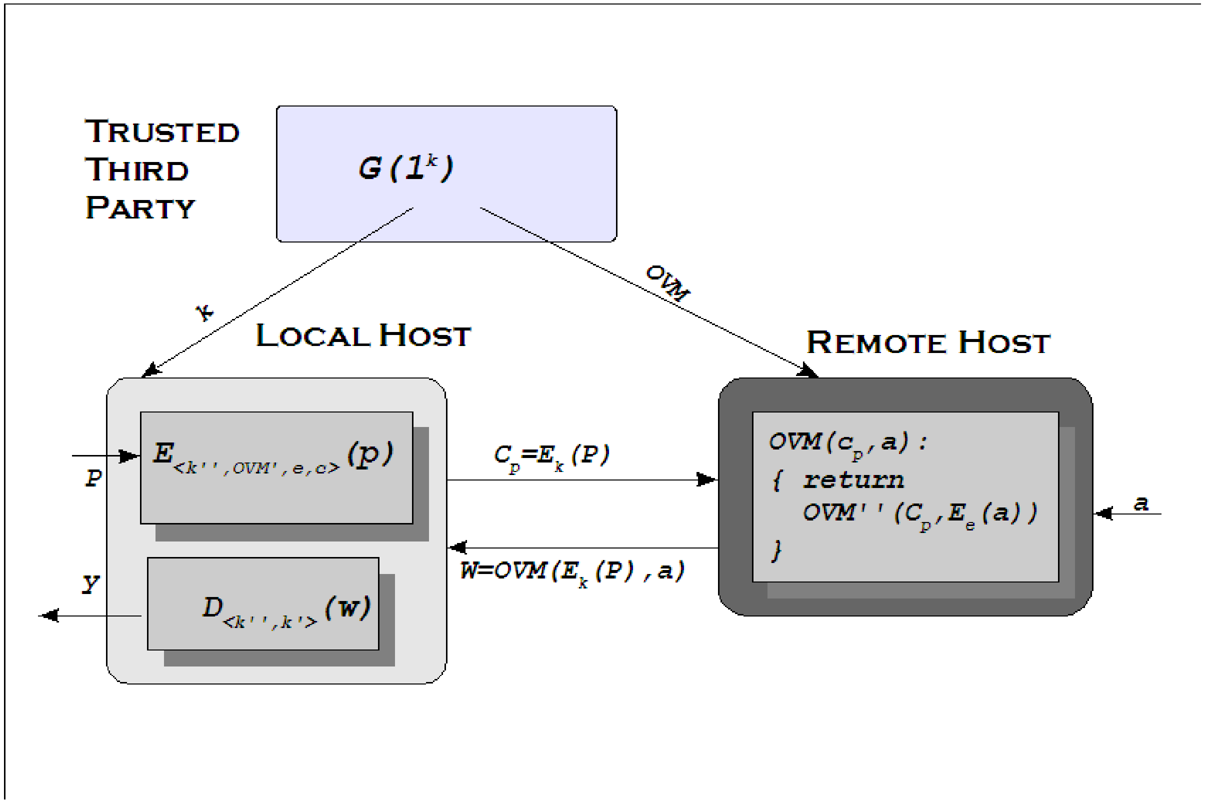
```
OVM(c_p,a):
{ return
    OVM''(C_p,E_e(a))
}
```

# WBRPE Cascade: Generation $G(1^k)$

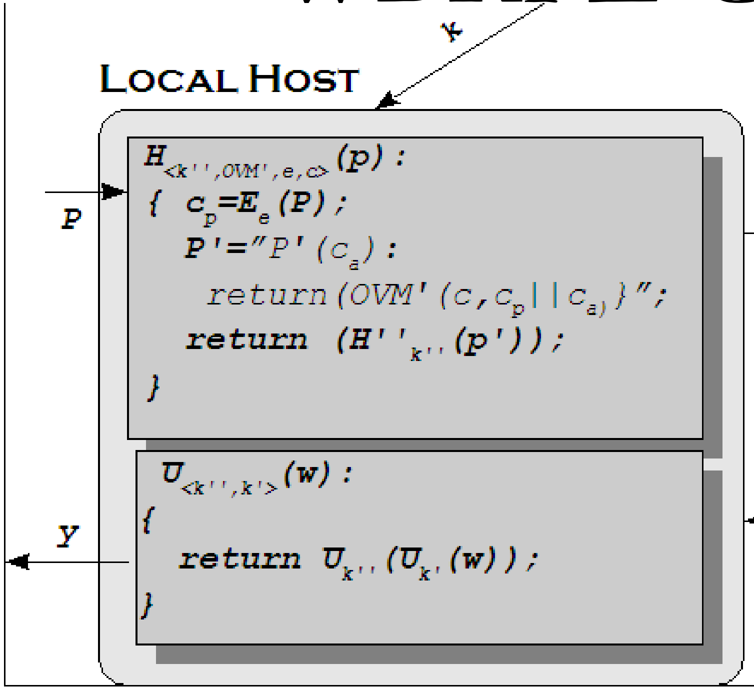


TRUSTED THIRD PARTY

```
G(1^k): { (k',OVM')←G'(1^k), (k'',OVM'')←G''(1^k), (e,d)←G_E(1^k);
          OVM="OVM(C_p',a):{return OVM''(C_p',E_e(a))}";
          UP="UP(c_p||c_a):{p=D_d(c_p); return P(D_d(c_a) }";
          c_up=H'_k'(UP); k=(k'',k',OVM',e,c_up);
          Return (k,OVM);
        }
```

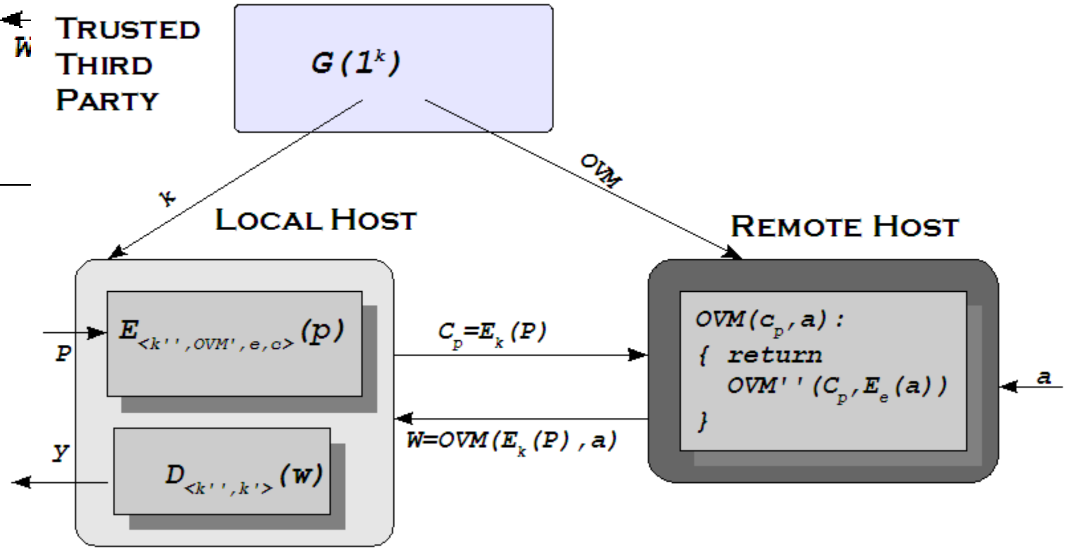


# WBRPE Cascade: Local Host



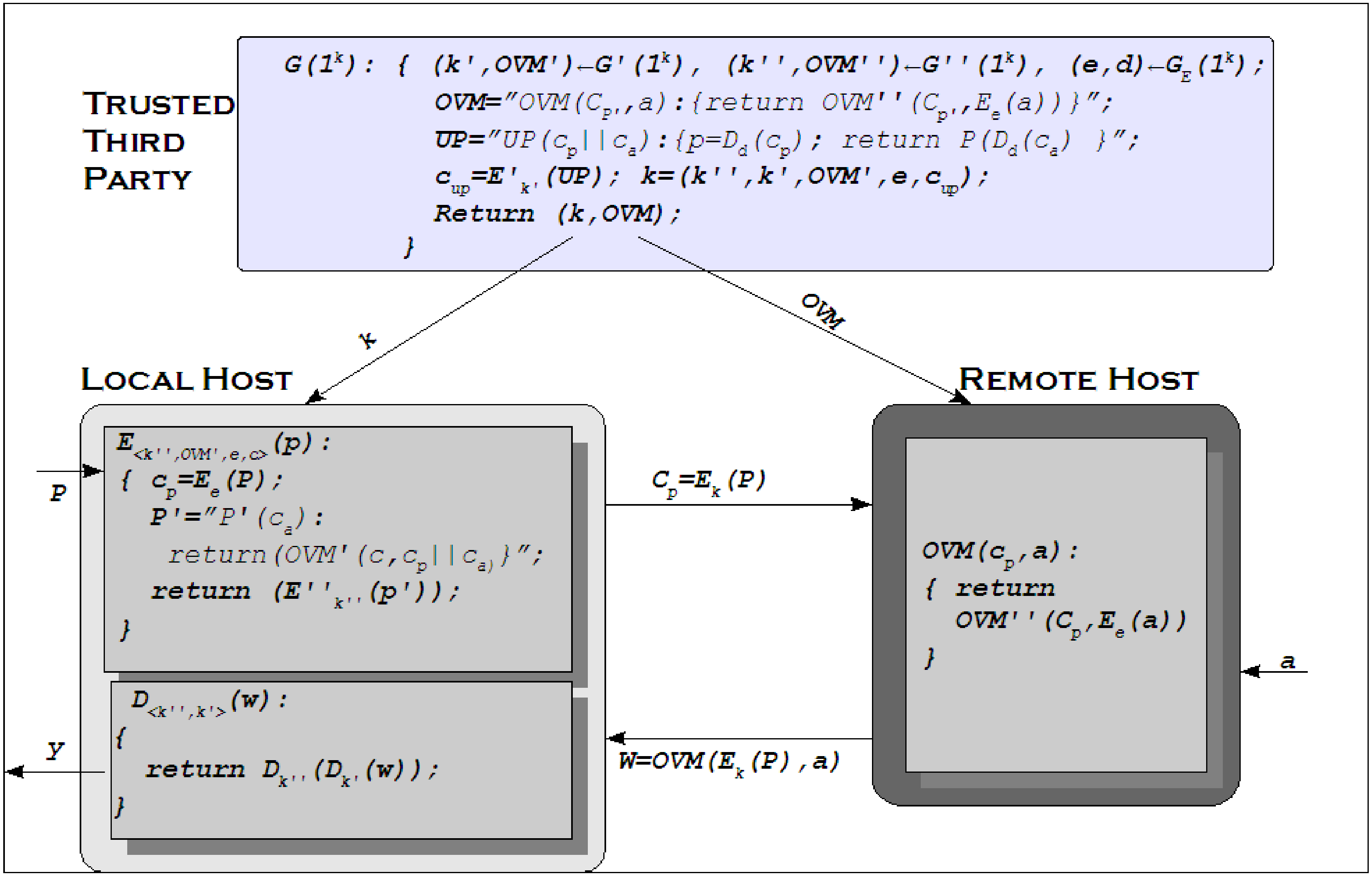
```
$H_{<k'',OVM',e,c>}(p)$:
{ $c_p = E_e(P)$;
  P'="P'($c_a$):
   return(OVM'(c,$c_p$||$c_{a)}$}";
  return ($H''_{k''}(p')$);
}
```

```
$U_{<k'',k'>}(w)$:
{
  return $U_{k''}(U_{k'}(w))$;
}
```



```
OVM($c_p$,a):
{ return
  OVM''($C_p$,$E_e$(a))
}
```

# WBRPE Cascade: a Robust Combiner



TRUSTED THIRD PARTY

```
$G(1^k)$: { $(k',OVM') \leftarrow G'(1^k)$, $(k'',OVM'') \leftarrow G''(1^k)$, $(e,d) \leftarrow G_E(1^k)$;
        $OVM$="$OVM(C_{p'},a)$:{return $OVM''(C_{p'},E_e(a))$}";
        $UP$="$UP(c_p||c_a)$:{$p=D_d(c_p)$; return $P(D_d(c_a)$ }";
        $c_{up}=E'_{k'}(UP)$; $k=(k'',k',OVM',e,c_{up})$;
        Return $(k,OVM)$;
      }
```

LOCAL HOST

```
$E_{<k'',OVM',e,c>}(p)$:
{ $c_p=E_e(P)$;
   $P'$="$P'(c_a)$:
    return$(OVM'(c,c_p||c_a)\}$";
   return $(E''_{k''}(p'))$;
}
```

```
 $D_{<k'',k'>}(w)$:
{
  return $D_{k''}(D_{k'}(w))$;
}
```

REMOTE HOST

```
$OVM(c_p,a)$:
{ return
  $OVM''(C_p,E_e(a))$
}
```

# Conclusions and Further Work

- Goal: foundations to white box security
- WBRPE: alternative model for SW 'hardening'
  - Candidate for `white box security building block`
- Presented Robust Combiner for WBRPE
  - Secure if at least one of the candidates is secure
  - Some details skipped (esp.: program validation)
- Questions?
- Thank you