1

2

JNCF 2017

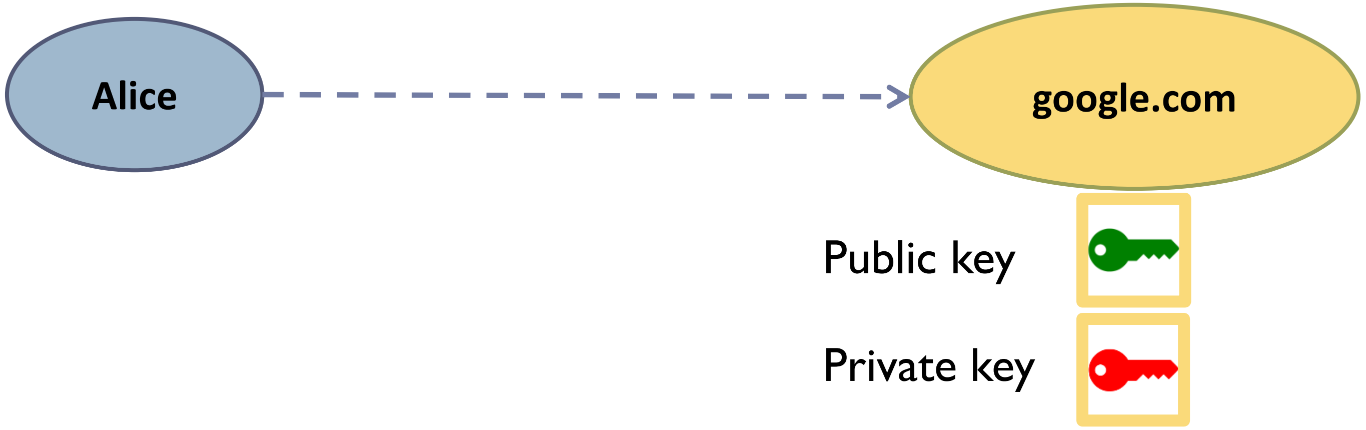
2017/01/20

Private Multi-party Matrix Multiplication and Trust Computations

Jean-Guillaume Dumas[1] ; Pascal Lafourcade[2] ; Jean-Baptiste Orfila[1] ; Maxime Puys[1]

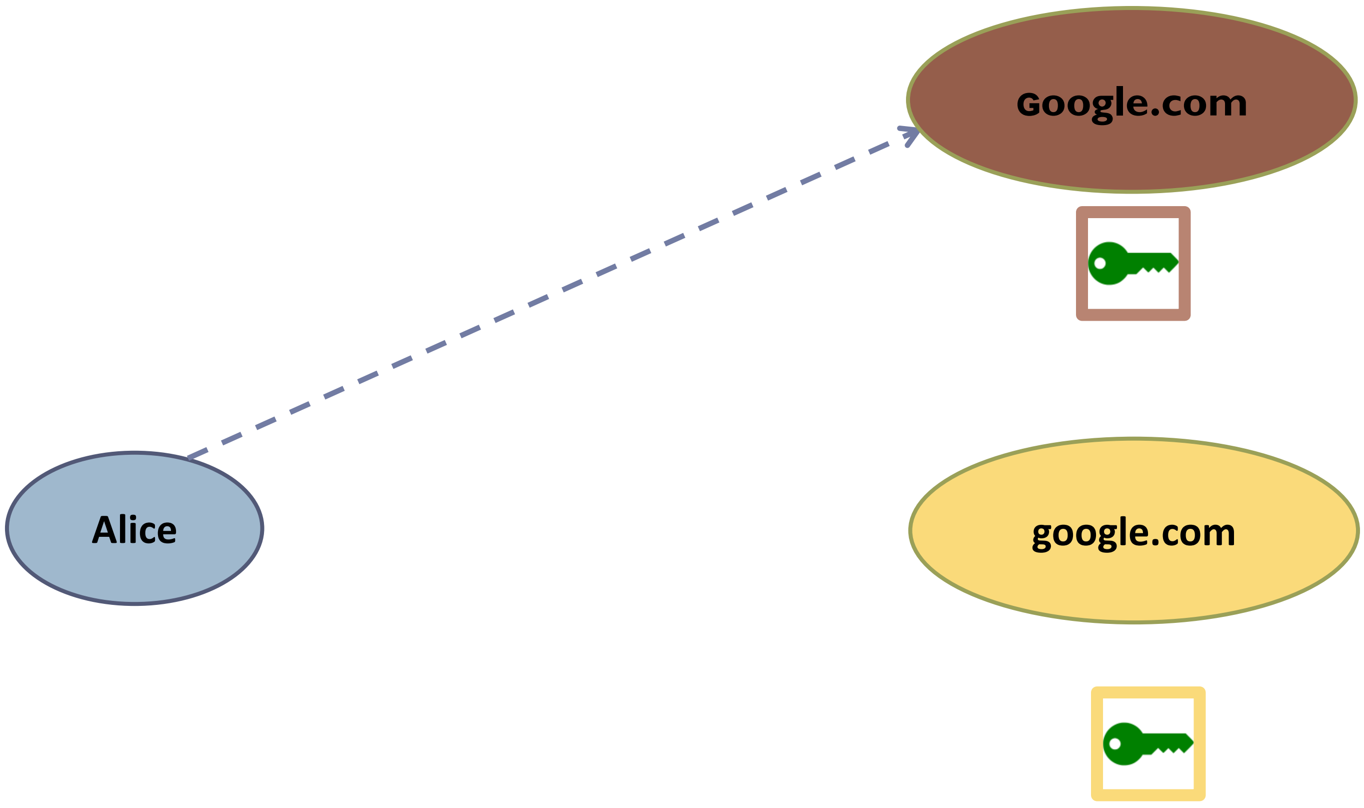
Public Key Infrastructure

- Alice wants to securely reach a website
  (e.g. using « https »)



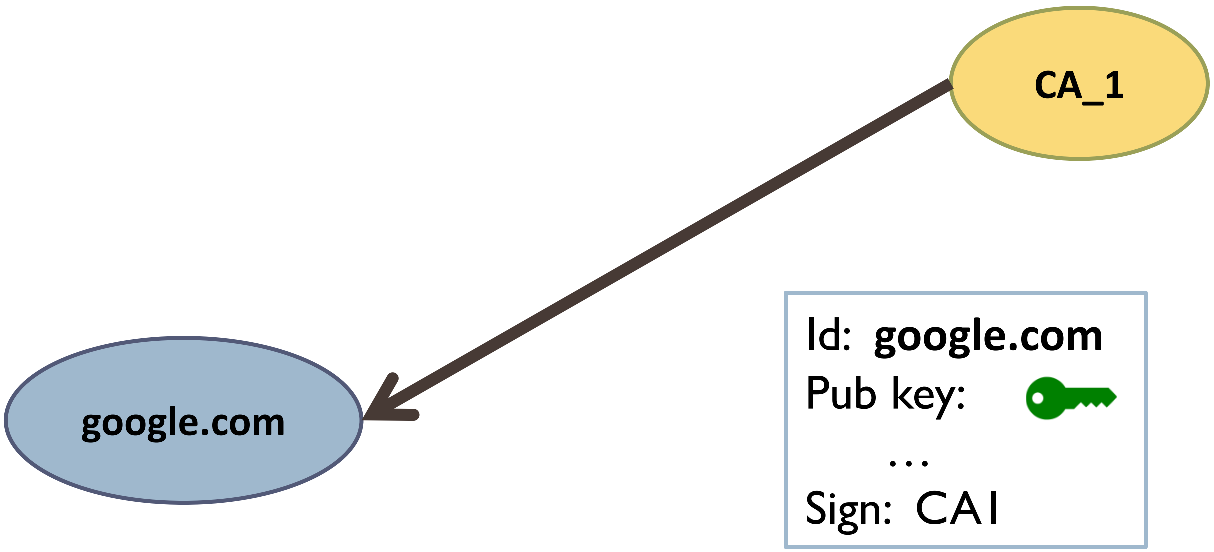
Public Key Infrastructure

- Problem : Fake website !



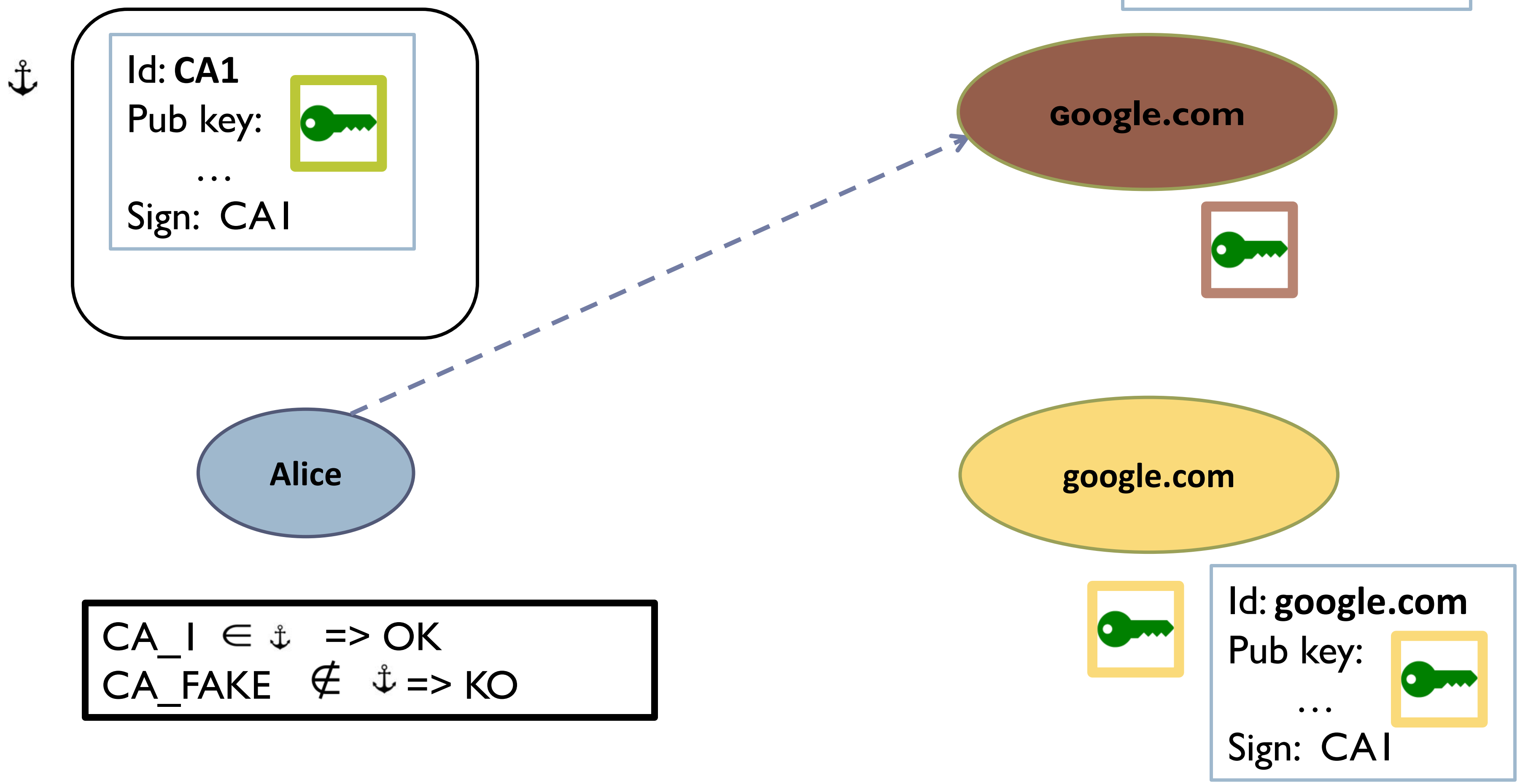
Public Key Infrastructure

- Certificates are delivered by a <u>certification authority (CA)</u>

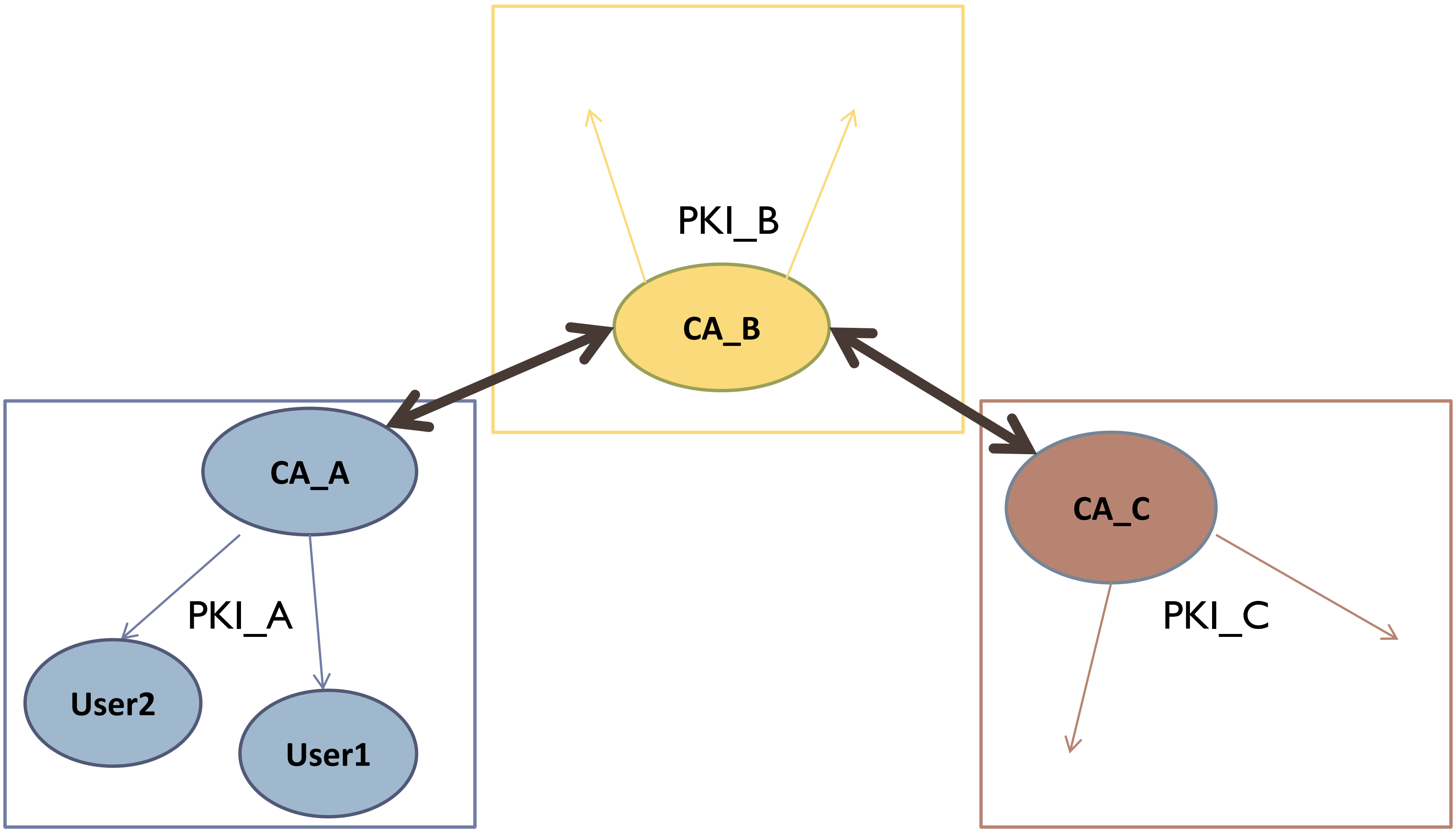


Public Key Infrastructure

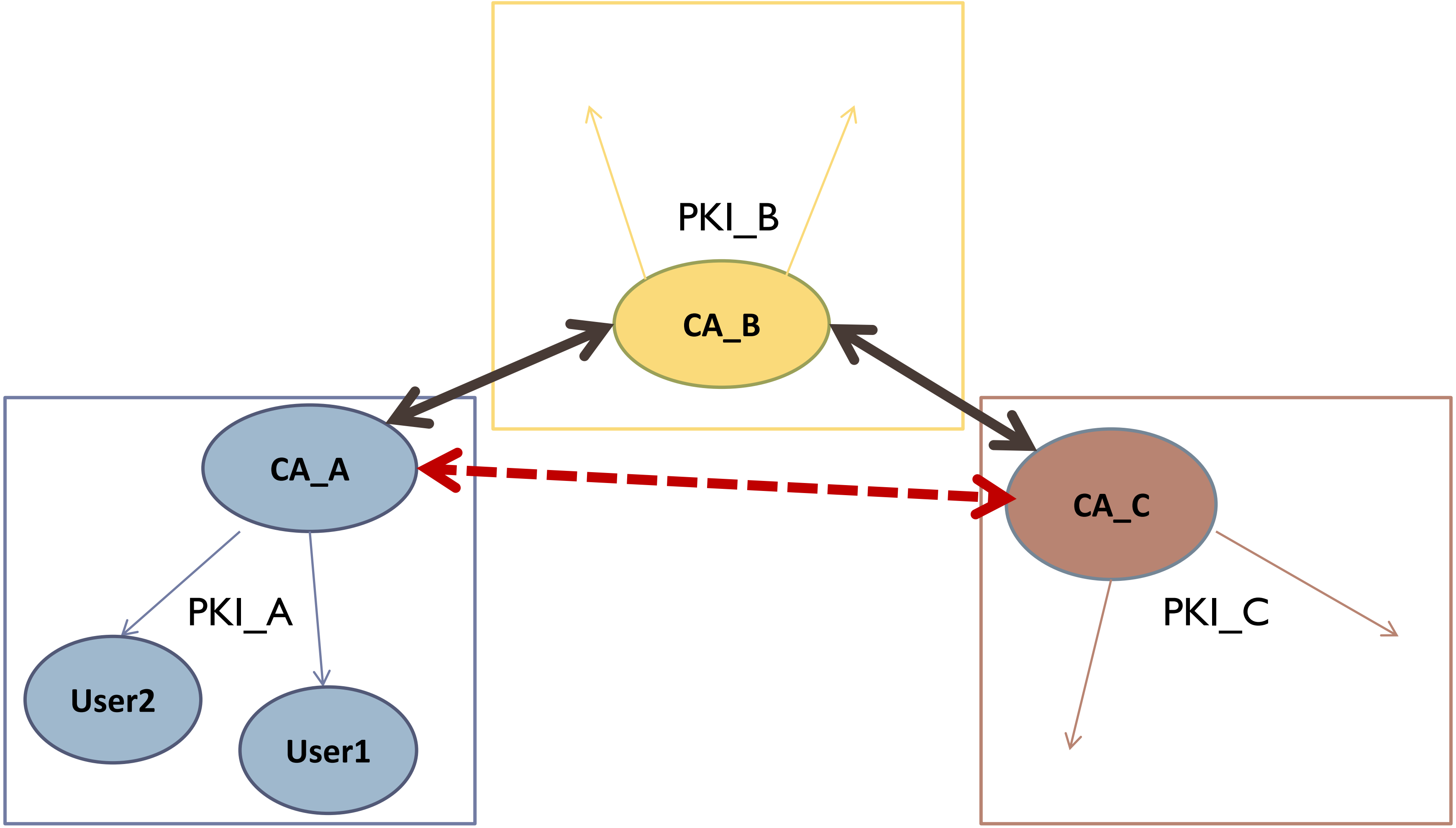
- Alice checks the certificate



Trust between CA

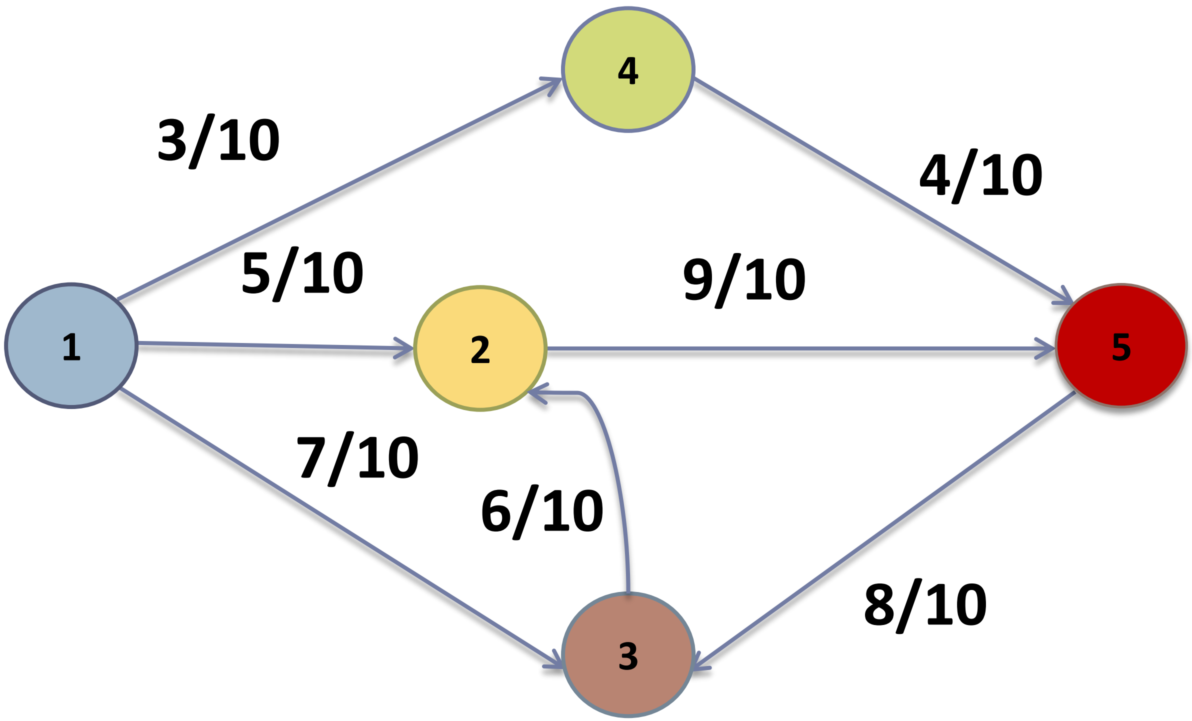


Trust between CA



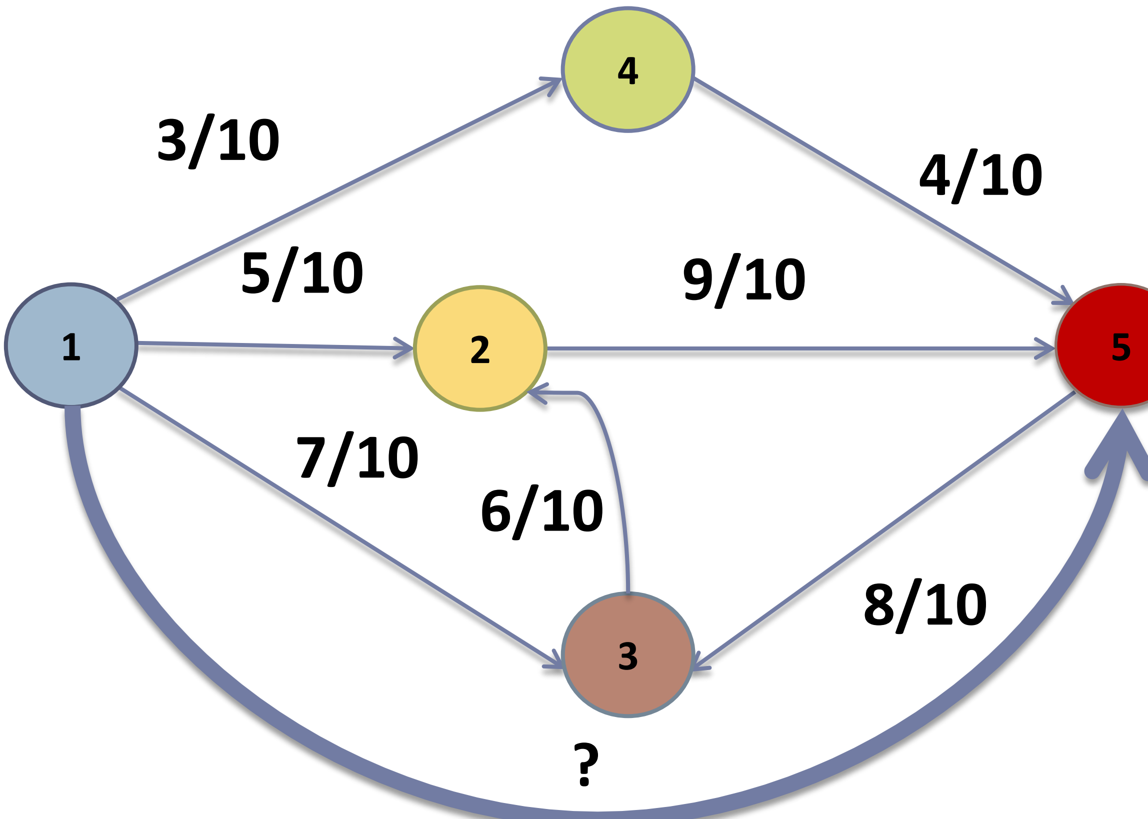
Network trust evaluation

- Trust value between nodes



Network trust evaluation

- Trust evaluation between P1 and P5 ?



Trust Model [Jøsang 2007]

- Trust metric:
  - T = (Trust, Distrust, Uncertainty)

Positive Experiences

Negative Experiences

1 – Trust - Distrust

- Trust Aggregation:
  - Direct evaluation:

Trust Model [Jøsang 2007]

- Trust metric:
  - T = (Trust, Distrust, Uncertainty)

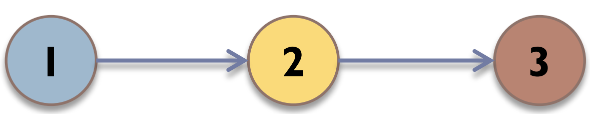
Positive Experiences

Negative Experiences

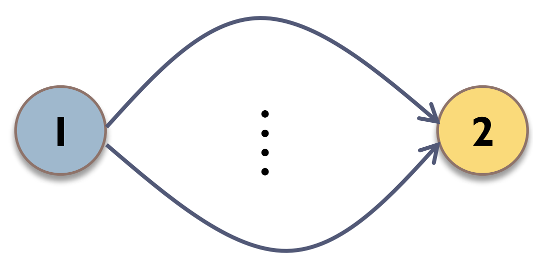
1 – Trust - Distrust

- Trust Aggregation (monoids based):

Sequential ('x')

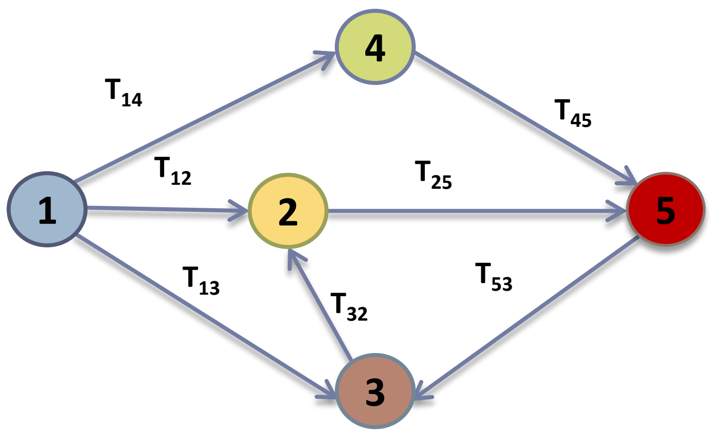


Parallel ('+')



Matrix representation

- From a graph…



…To a matrix

A =

	T_{12}	T_{13}	T_{14}	?
				T_{25}
	T_{32}			
				T_{45}
		T_{53}		

- Trust aggregation [Dumas, Hossayni, 2012]
  - k: longest path between vertices
  - A^k converges to global trust

Securely computing trust

- How to securely compute matrix product ?

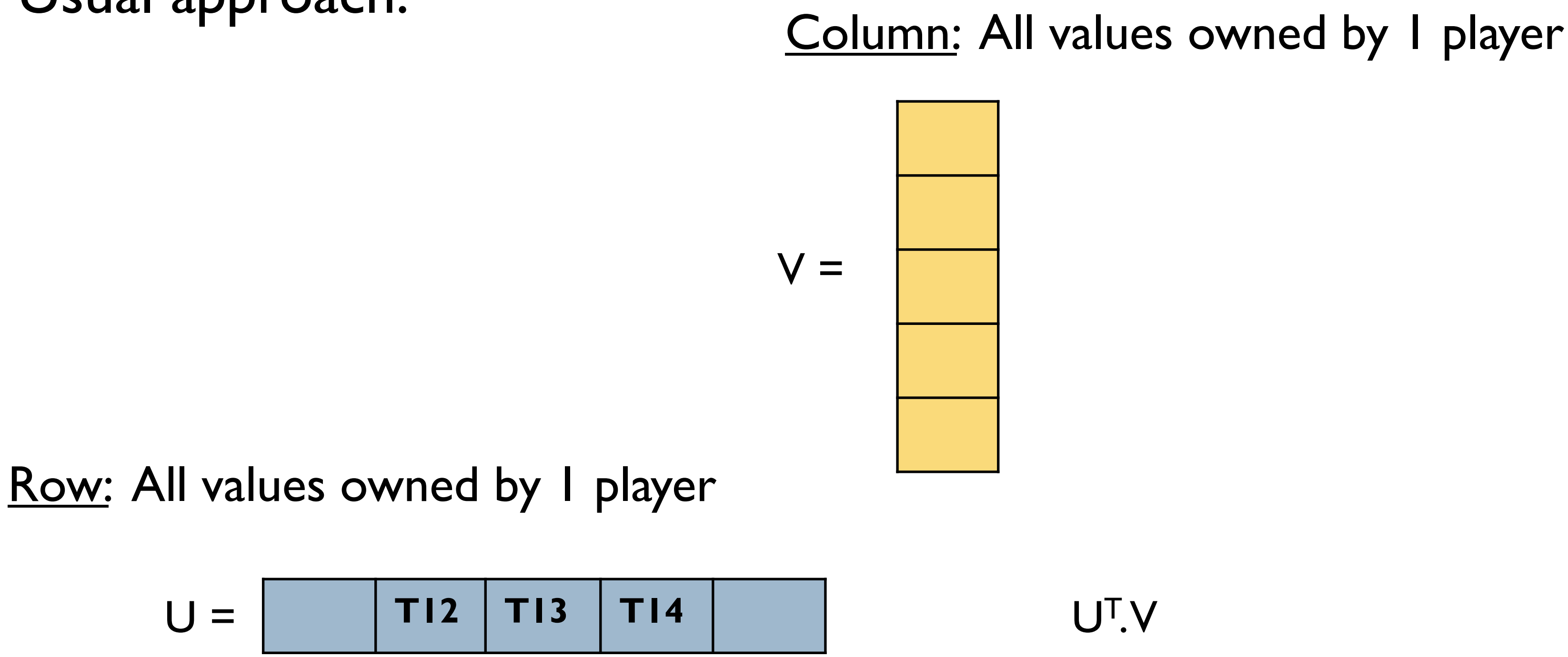
- Conditions:
  - *n* players
  - 1 secret input per player (i.e. the row)
  - 1 common computation (i.e. A^k)

Outline

1. Introduction
2. A secure multiparty dot product problem
   a. State of the art
   b. Definitions and tools
   c. Data repartition problem
3. A new dot product protocol
   DSDP
4. Security strenghtening of the DSDP protocol
   a. 1 player corruption
   b. Collusion attacks
   c. Random Ring Order
5. Conclusion

Secure dot product: State of the Art

- Usual approach:



- [Du et al. 2001]; [Amirbekyan et al. 2007]; [Wang et al. 2008];
- ...

Homomorphic Encryptions

- Homomorphic Encryptions:
  - $E_k(m1)\ E_k(m2) = E_k(m1+m2)$
  - $E_k(m1)^{m2} = E_k(m1.m2)$
- e.g. Cryptosystems of Paillier, Benaloh, Naccache-Stern…

- Paillier's cryptosystem:
  - ➕ Ciphering/Deciphering based on modular exponentiations (« RSA like »)
  - ➖ Cleartext space depends on each player's parameters

- Benaloh's cryptosystem:
  - ➖ Deciphering: computing an "easy" discrete log
  - ➕ Common cleartext space

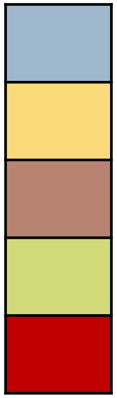
Dot product

- Data repartition:

Column: 1 secret value per player

$U^T.V$

V =



Row: All values owned by 1 player

U =

Security notions

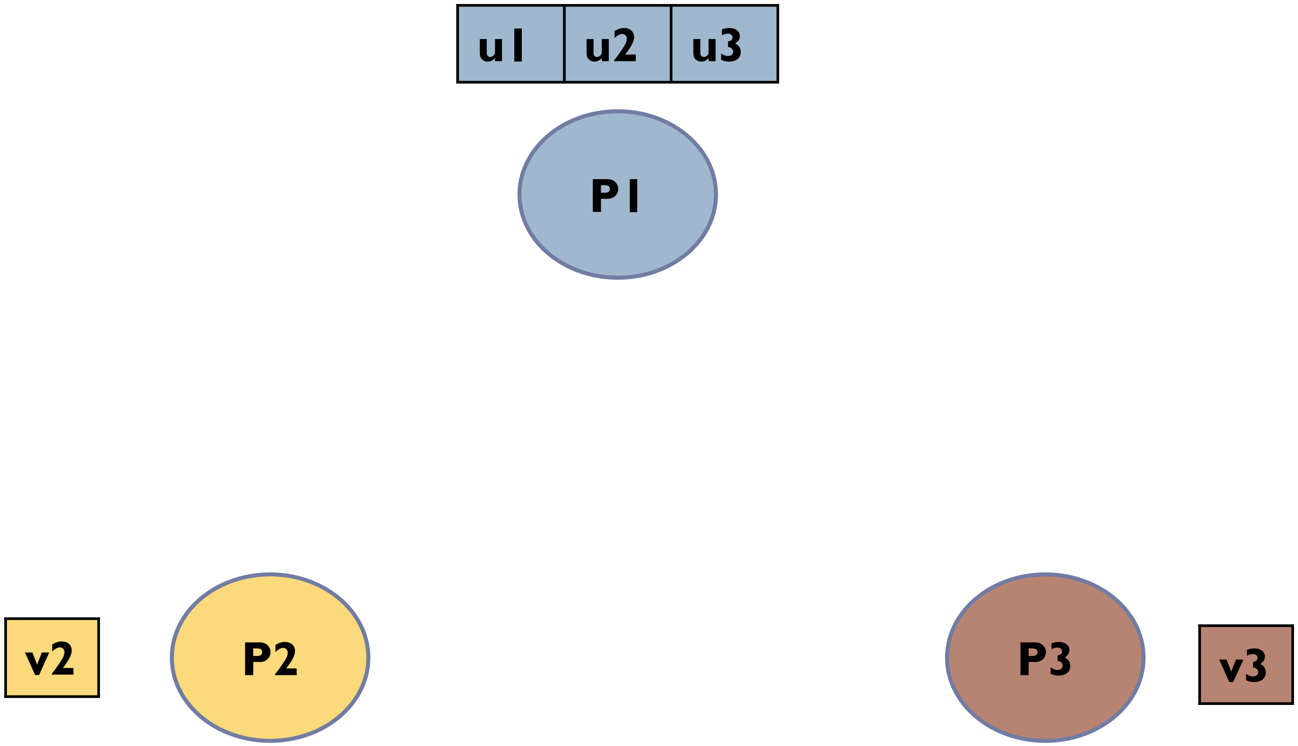
- Protocol must achieve…
  - Correctness
  - Privacy
  - Safety
- …despite adversaries…
  - Curious-but-honnest
  - Malicious
- …Capable of cooperating

State of the Art

- <u>MPWP</u>: [Dolev et al. '10]
  - Securely computing weighted average
  - Benaloh's cryptosystem
  - Communications cost: $O(n^3)$
- <u>P-MPWP</u>: (1st contribution)
  - Adaptation w/ Paillier's cryptosystem
  - Reduction of the communications: $O(n^2)$
- <u>DSDP</u>: (2nd contribution)
  - Paillier's cryptosystem
  - Communications cost: $O(n)$
  - Less security properties are verified

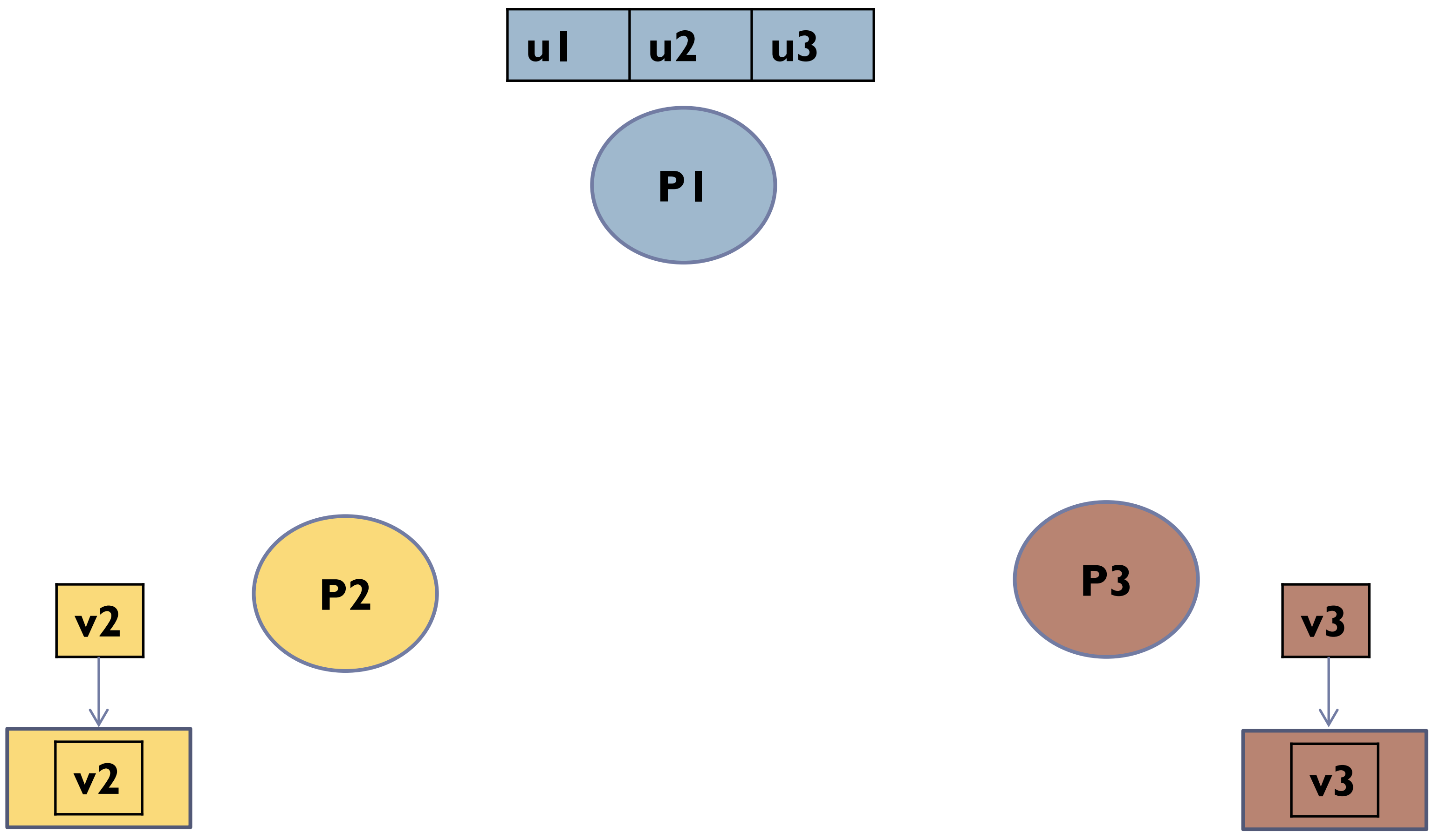
Distributed Secure Dot Product (DSDP)

- 0. Data repartition



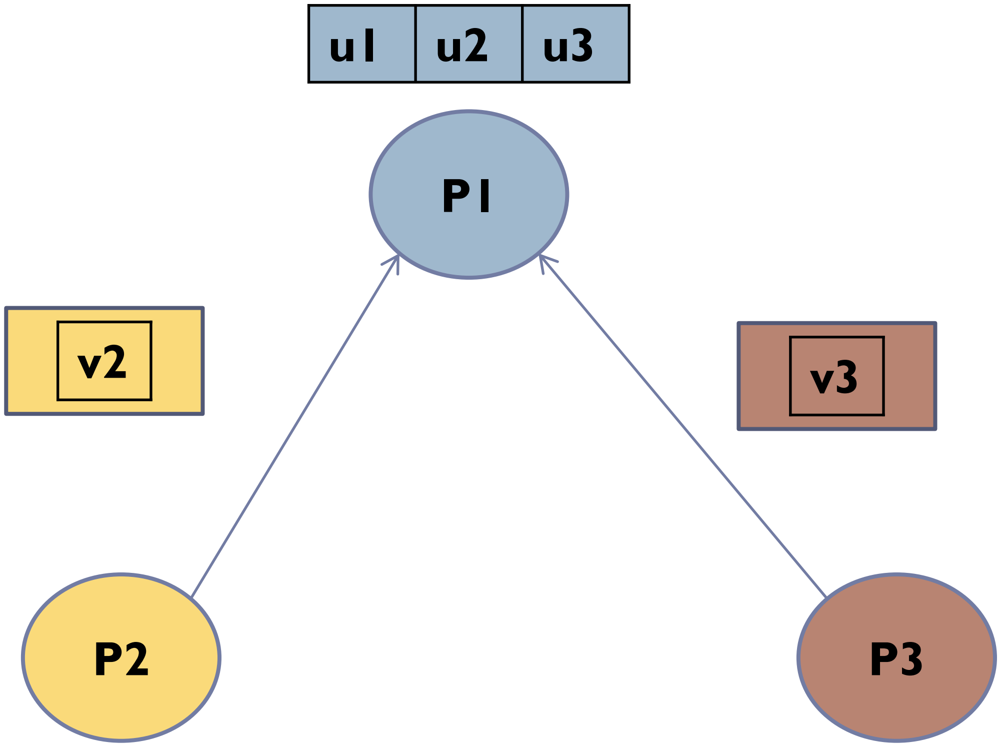
Distributed Secure Dot Product (DSDP)

- 1. Protection of P2 and P3 inputs -> ciphering



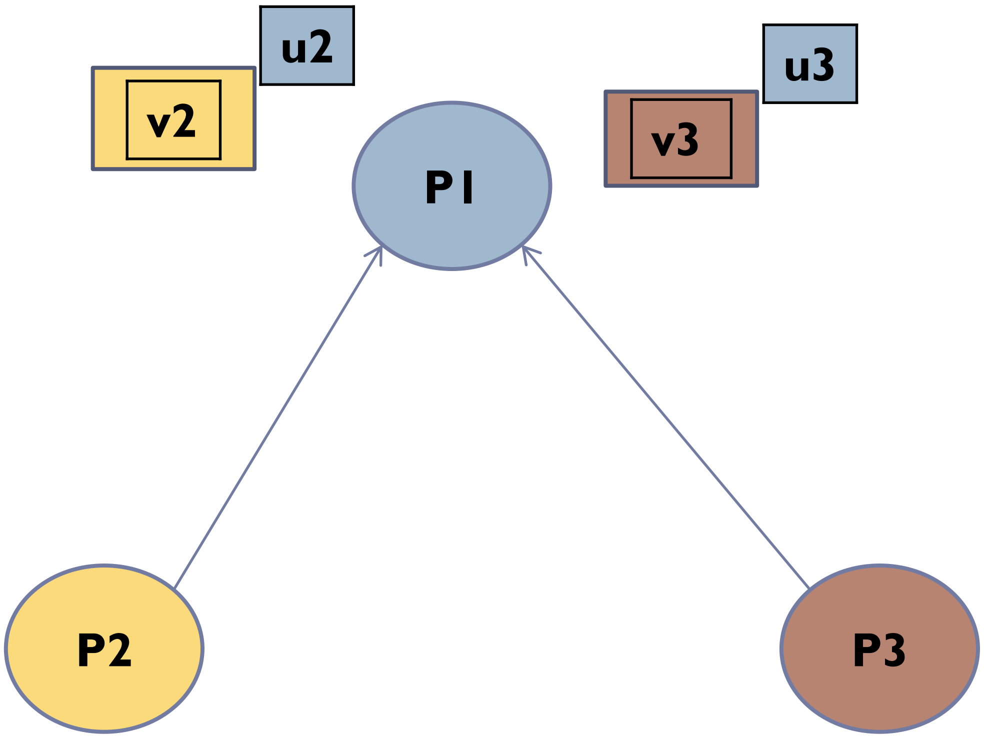
Distributed Secure Dot Product (DSDP)

- 2. Data exchange



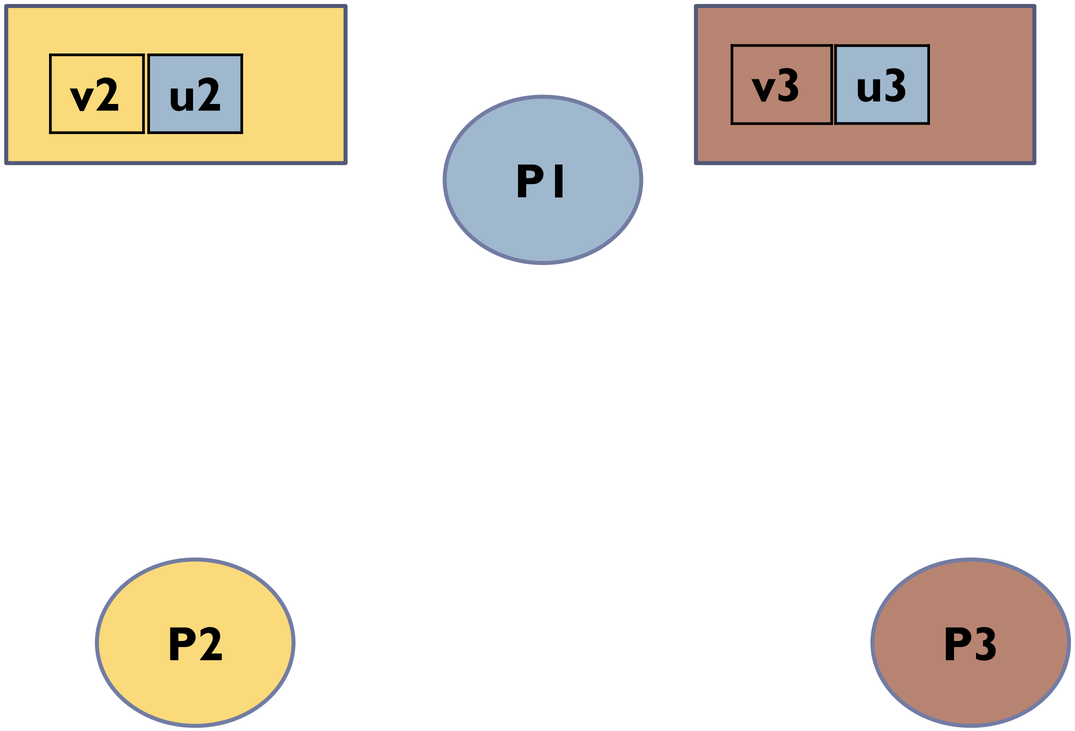
Distributed Secure Dot Product (DSDP)

- 3. Homomorphic operations



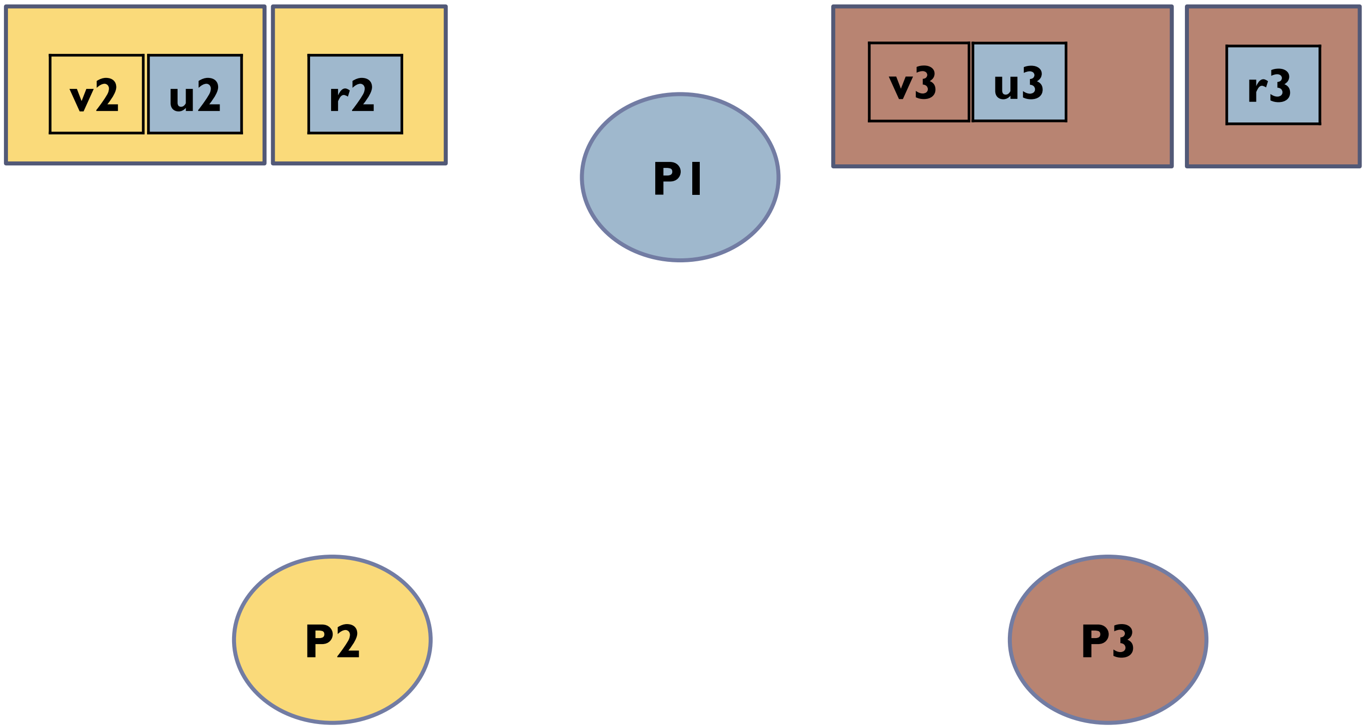
Distributed Secure Dot Product (DSDP)

- 3. Homomorphic operations



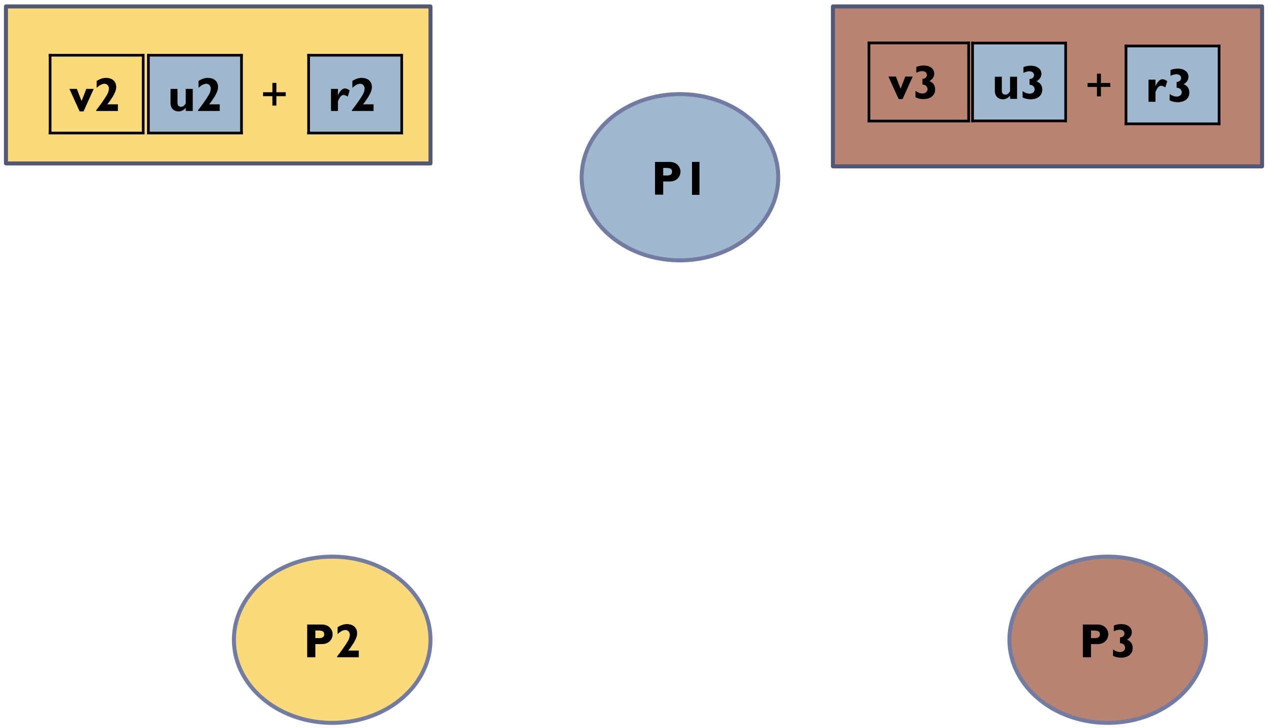
Distributed Secure Dot Product (DSDP)

- 4. P1 data protection: adding randomness



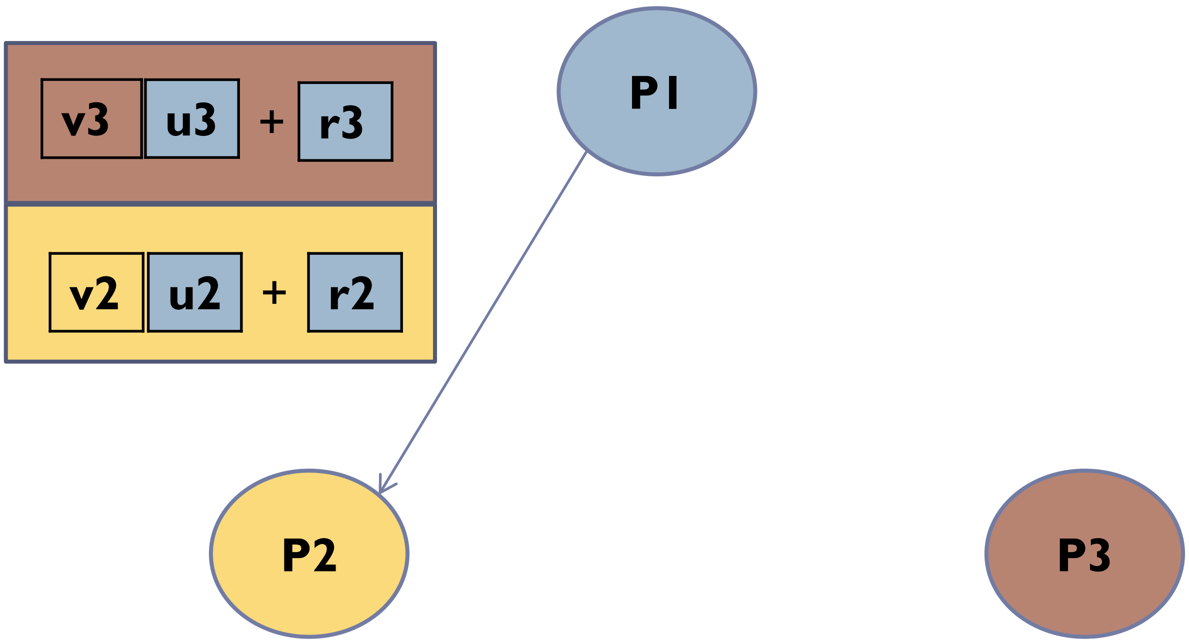
Distributed Secure Dot Product (DSDP)

- 4. P1 data protection: homomorphic operations



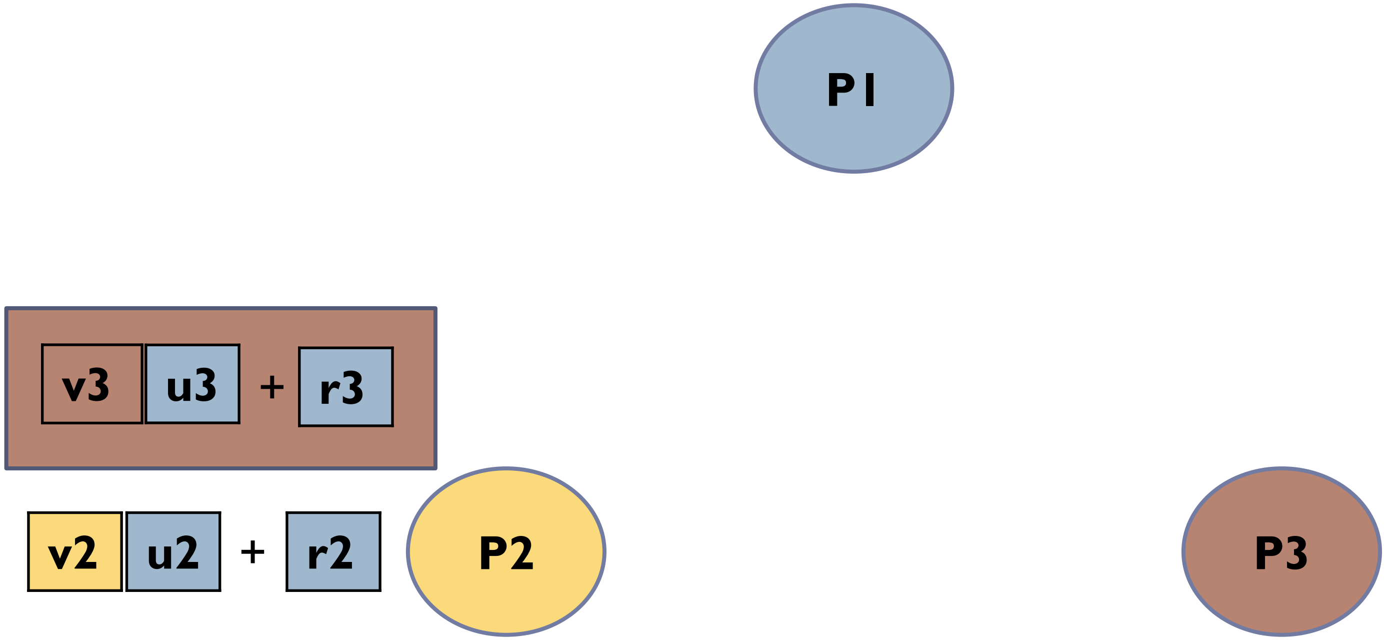
Distributed Secure Dot Product (DSDP)

- 5. Data exchange



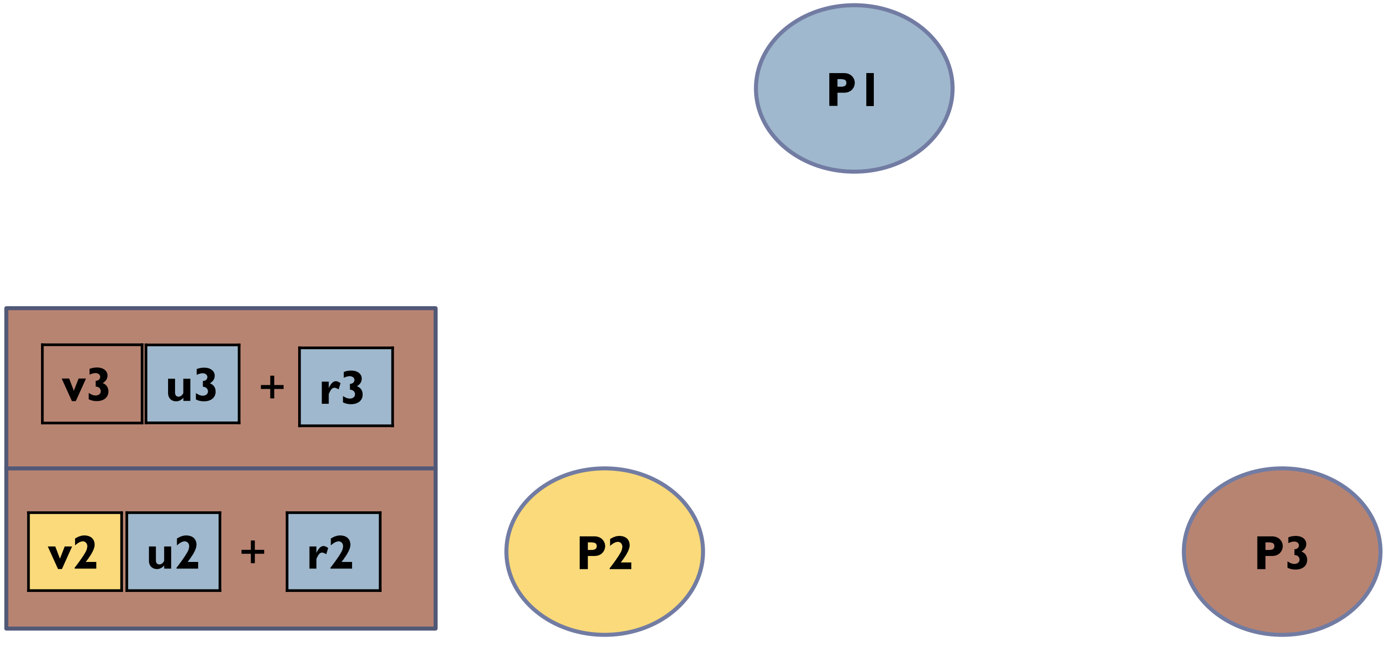
Distributed Secure Dot Product (DSDP)

- 6. Deciphering



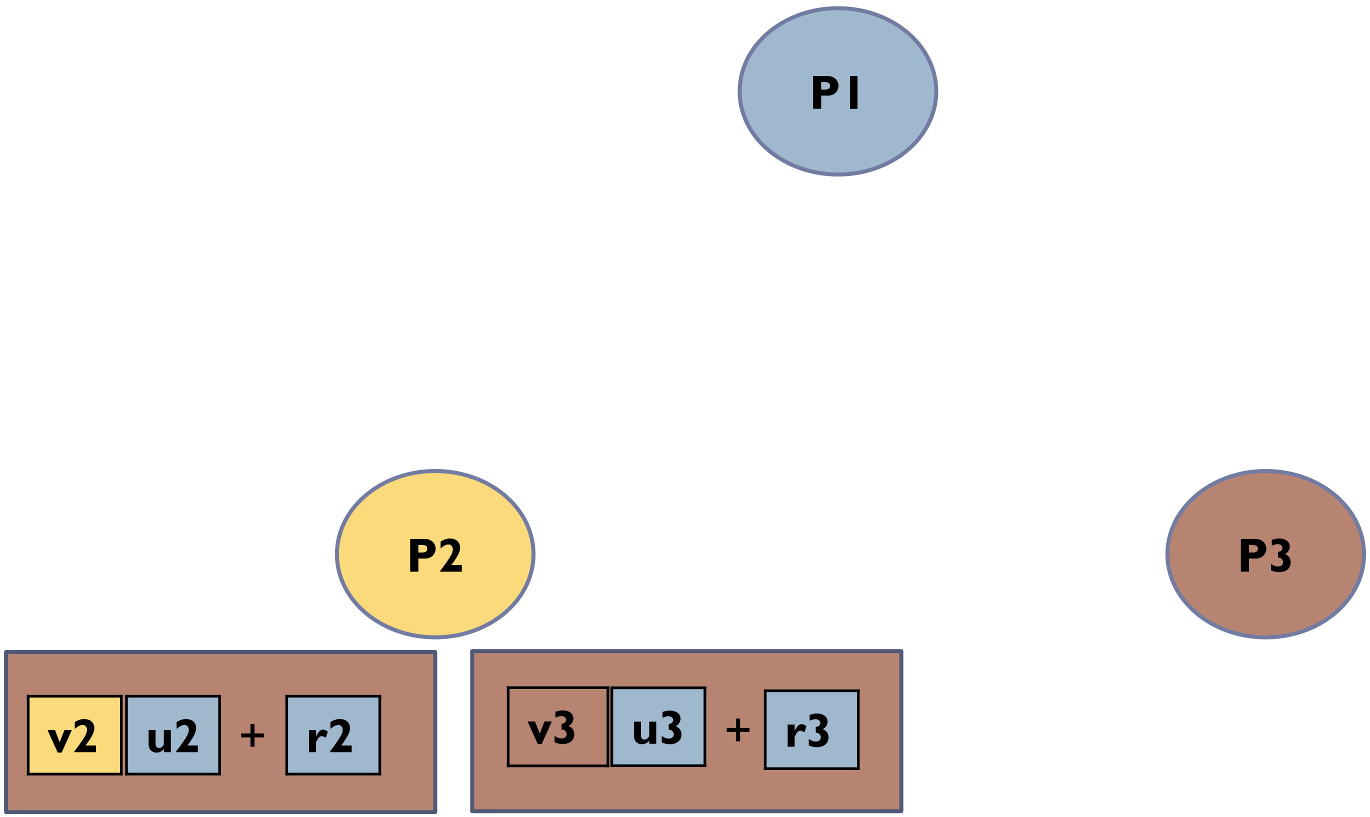
Distributed Secure Dot Product (DSDP)

- 7. Reciphering with next player's key



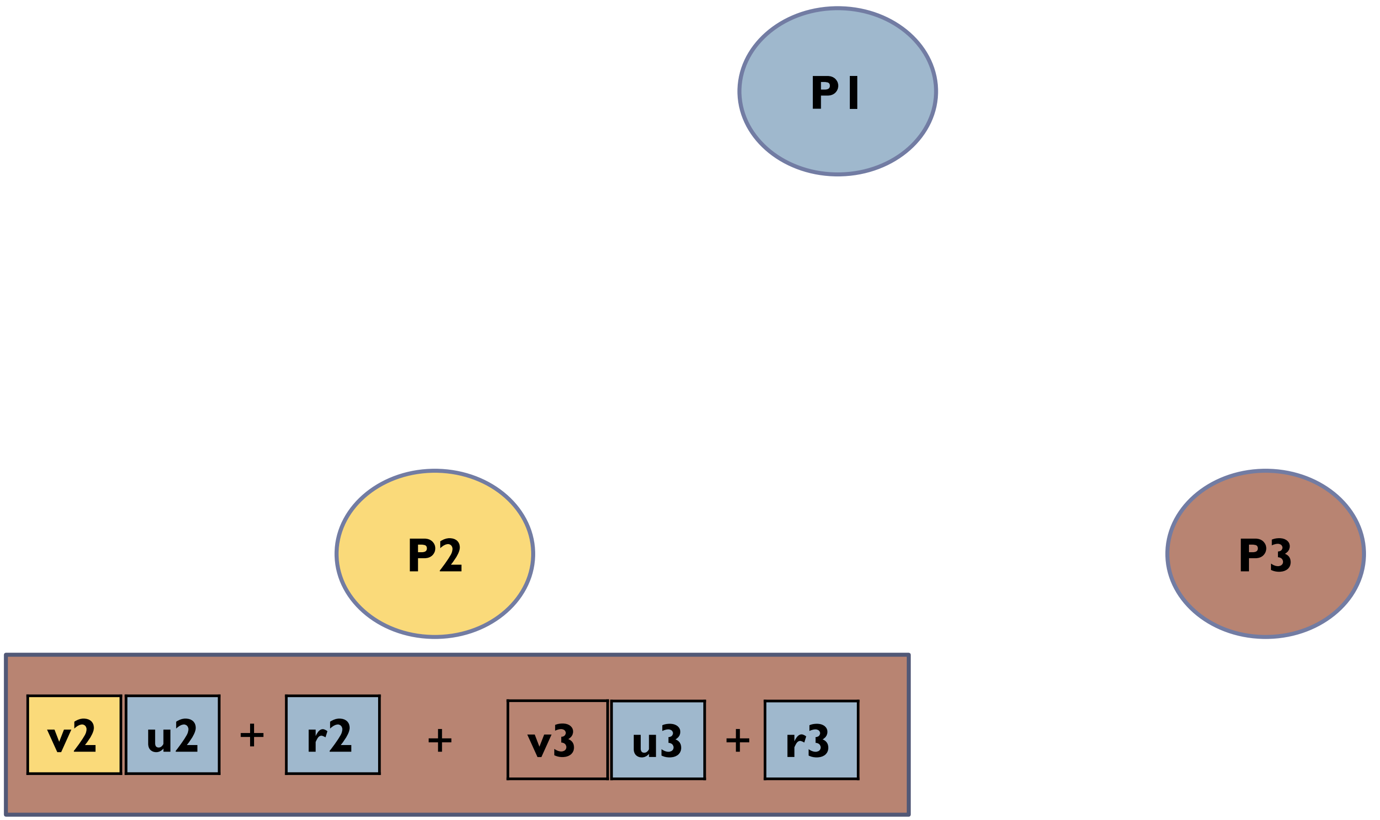
Distributed Secure Dot Product (DSDP)

- 8. Homomorphic operation



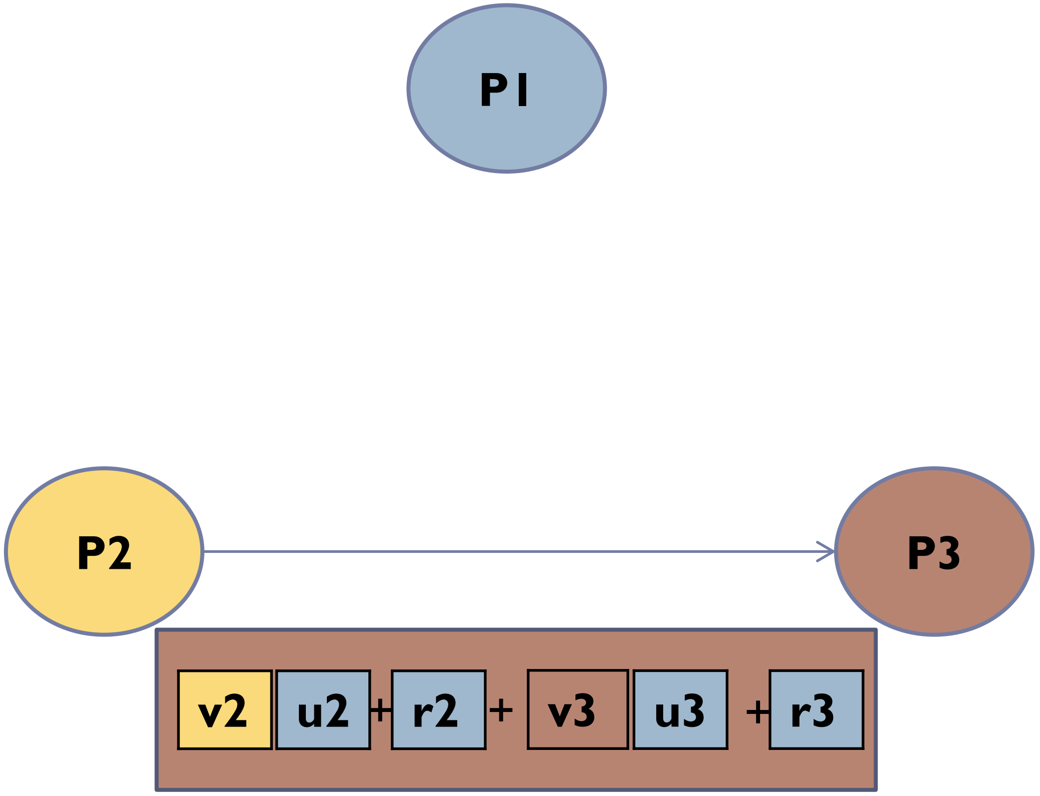
Distributed Secure Dot Product (DSDP)

- 8. Homomorphic operation



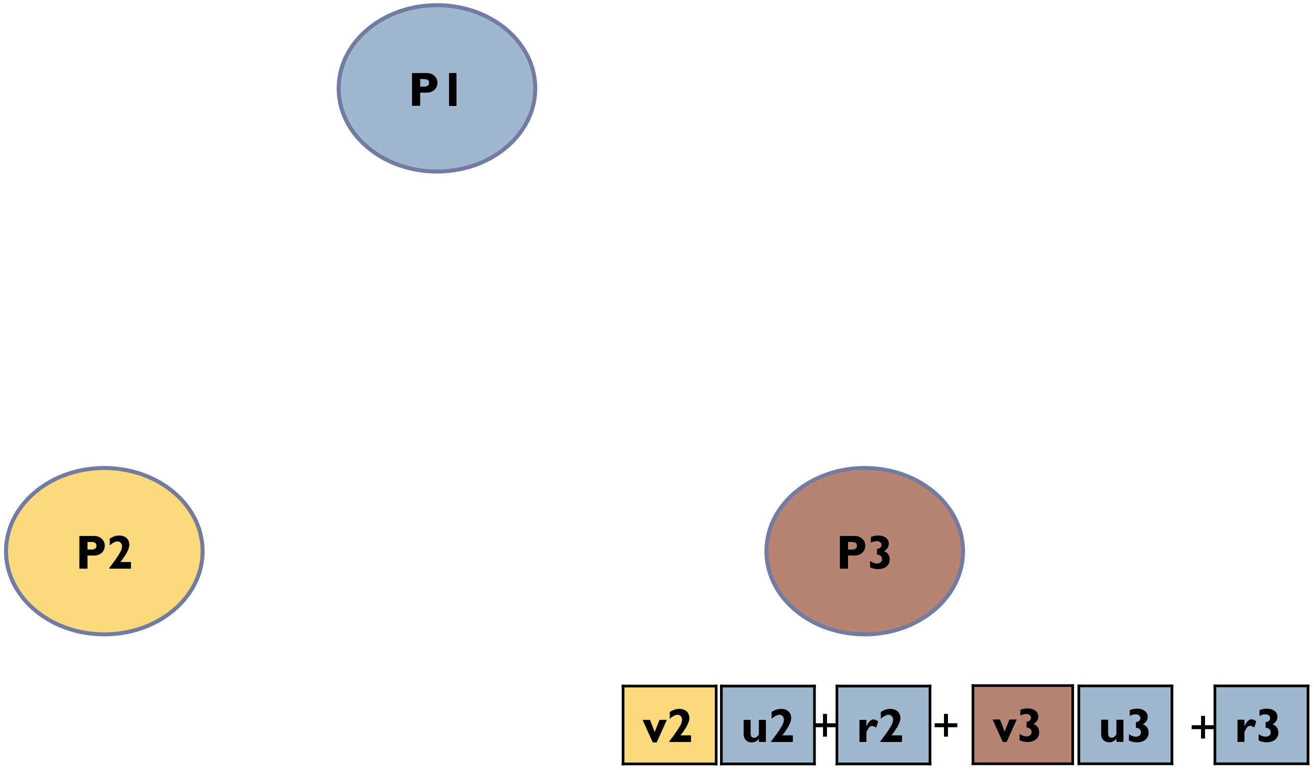
Distributed Secure Dot Product (DSDP)

- 9. Data exchange



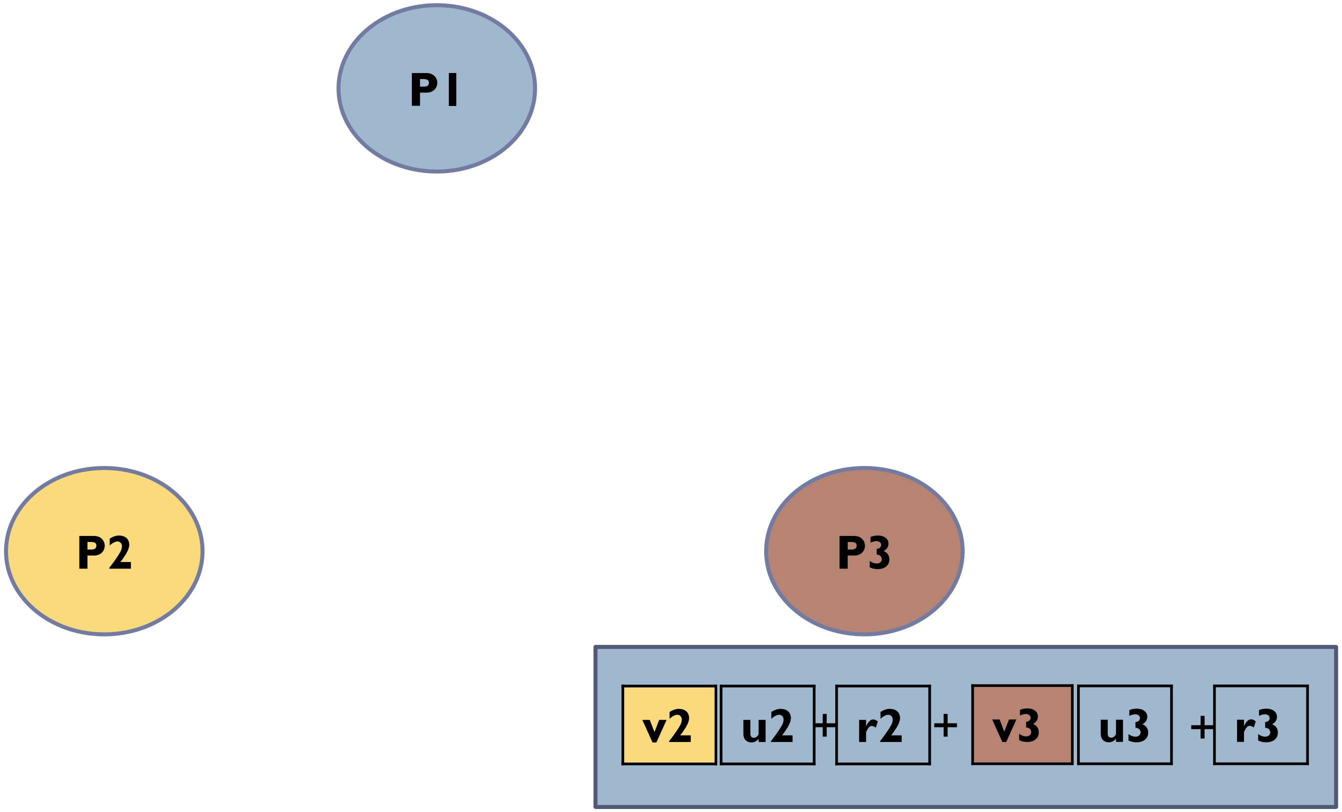
Distributed Secure Dot Product (DSDP)

- 10. Deciphering



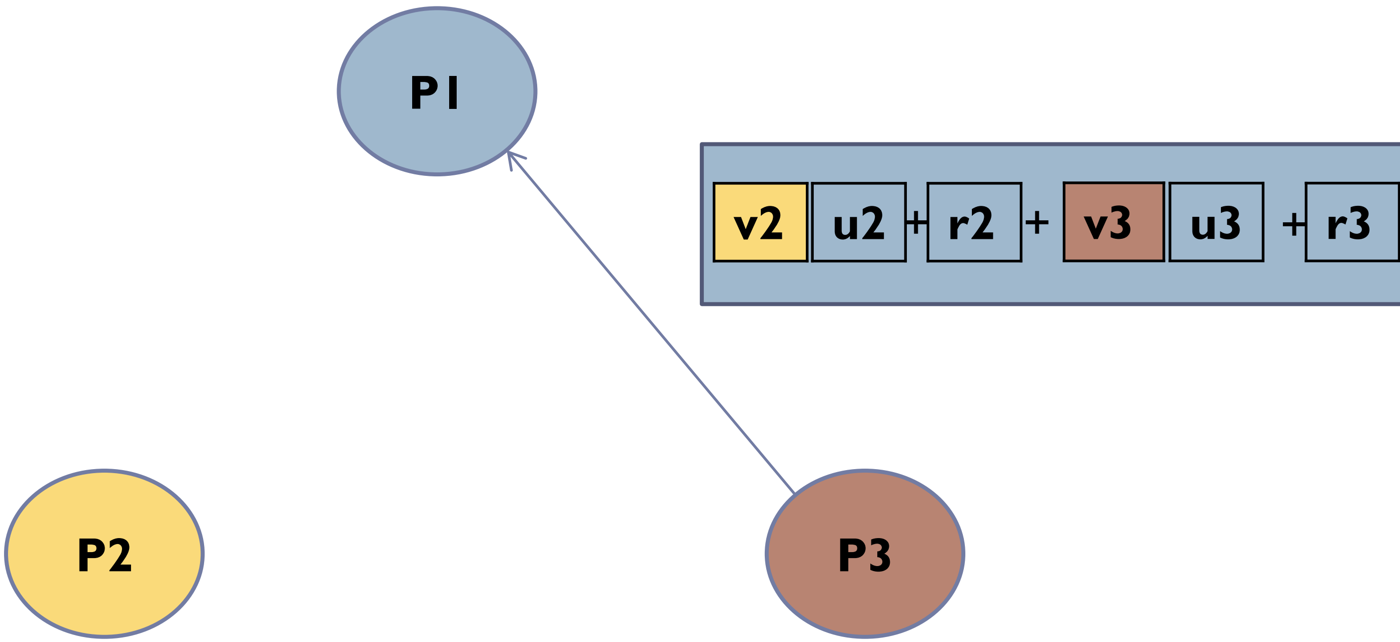
Distributed Secure Dot Product (DSDP)

- 11. Reciphering with master player's key



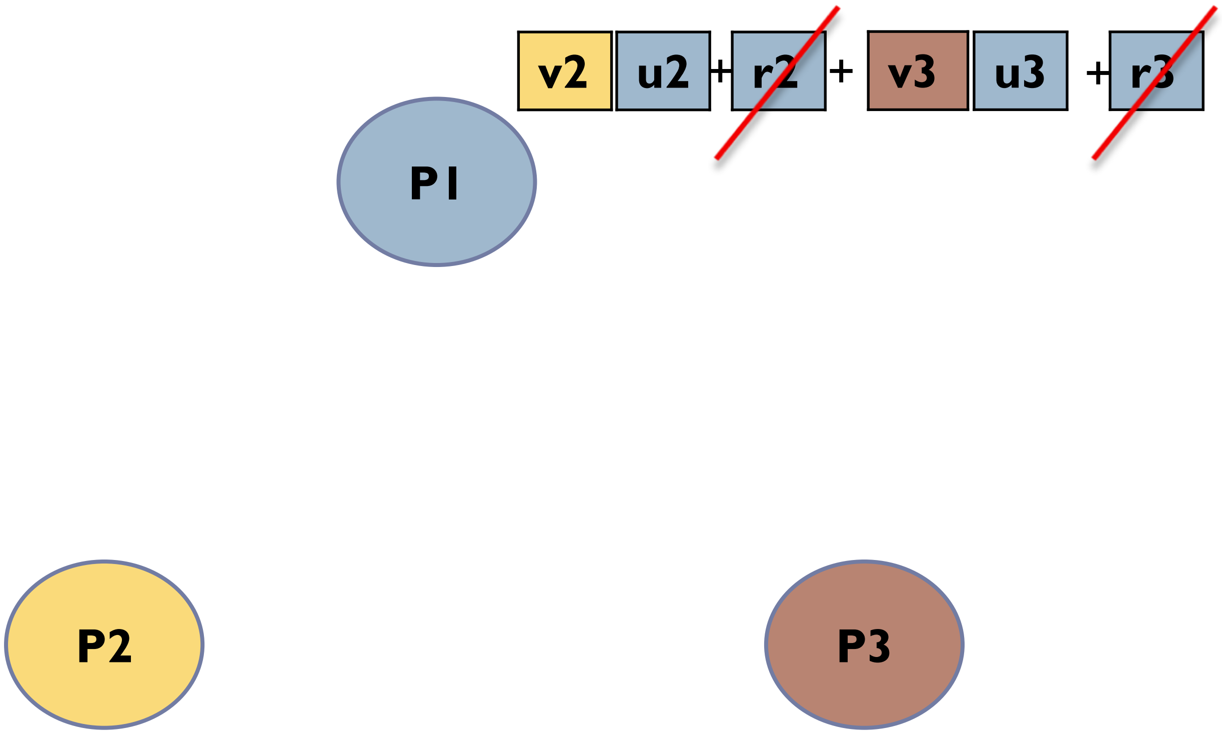
Distributed Secure Dot Product (DSDP)

- 12. Data exchange



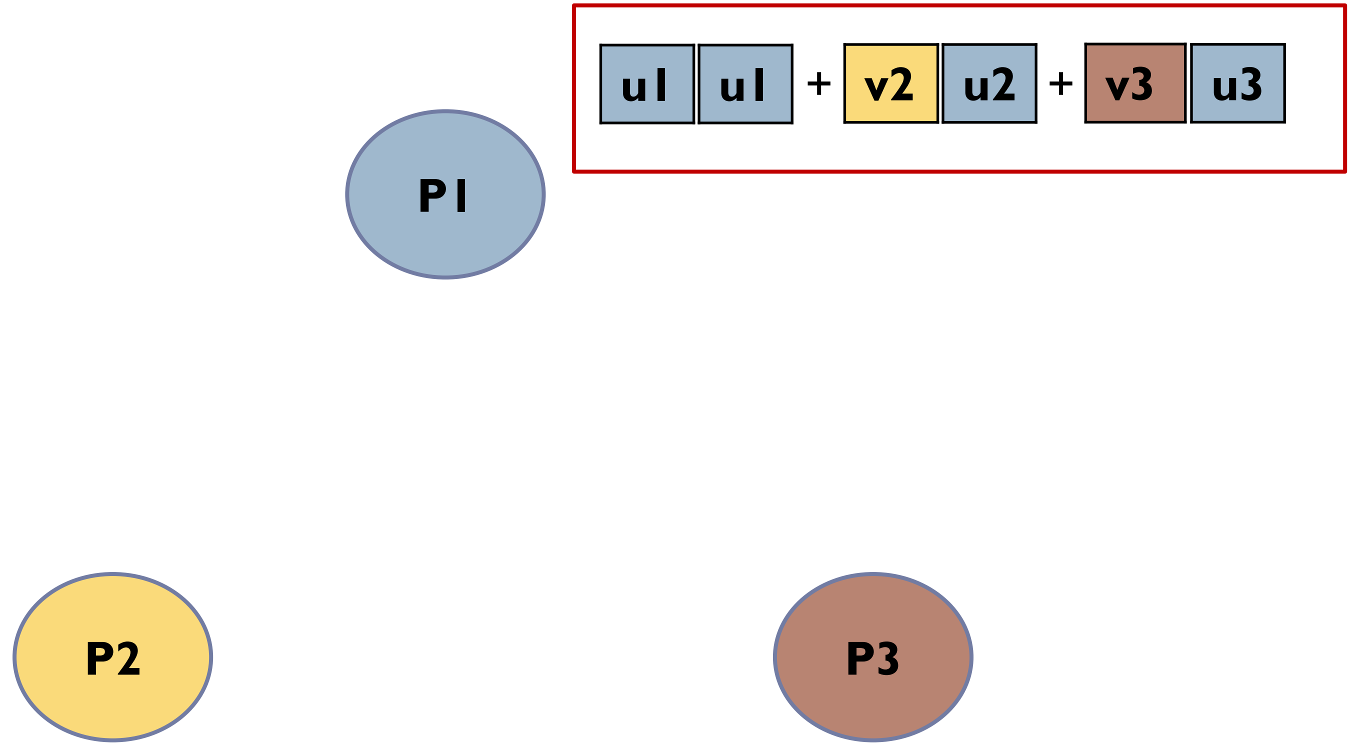
Distributed Secure Dot Product (DSDP)

- 13. Removing randomness



Distributed Secure Dot Product (DSDP)

- 14. Adding missing data

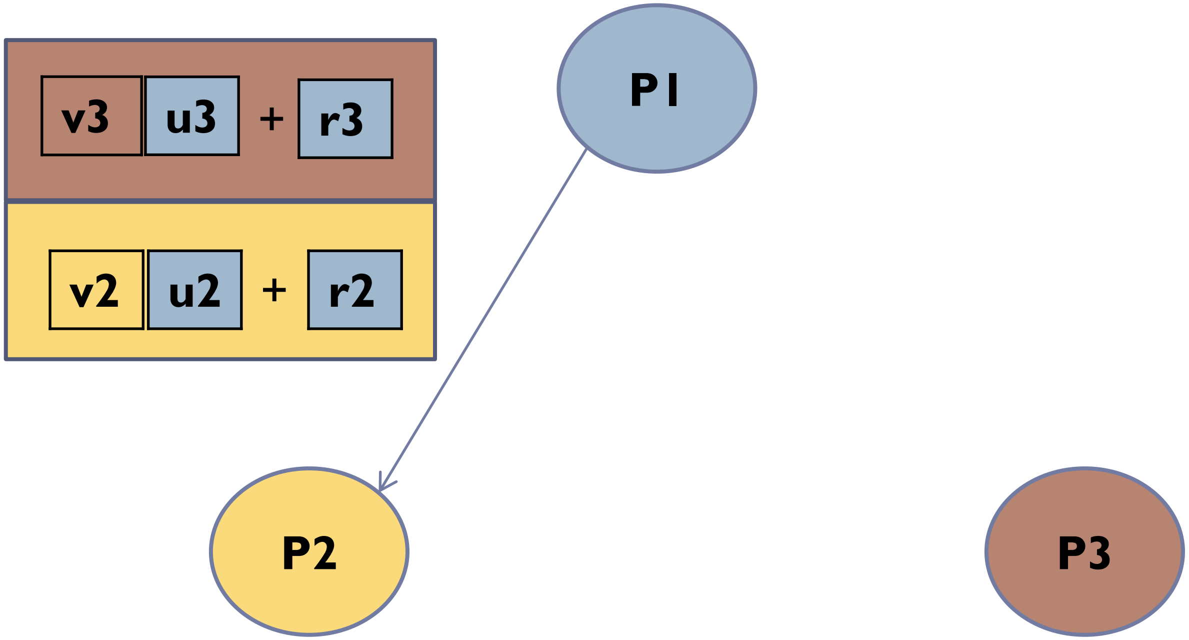


Distributed Secure Dot Product (DSDP)

- Properties:
  - Correctness
  - Security against one semi-honest adversary
  - Safety
  - O(n) communications
- Automatic security verification
  - ProVerif

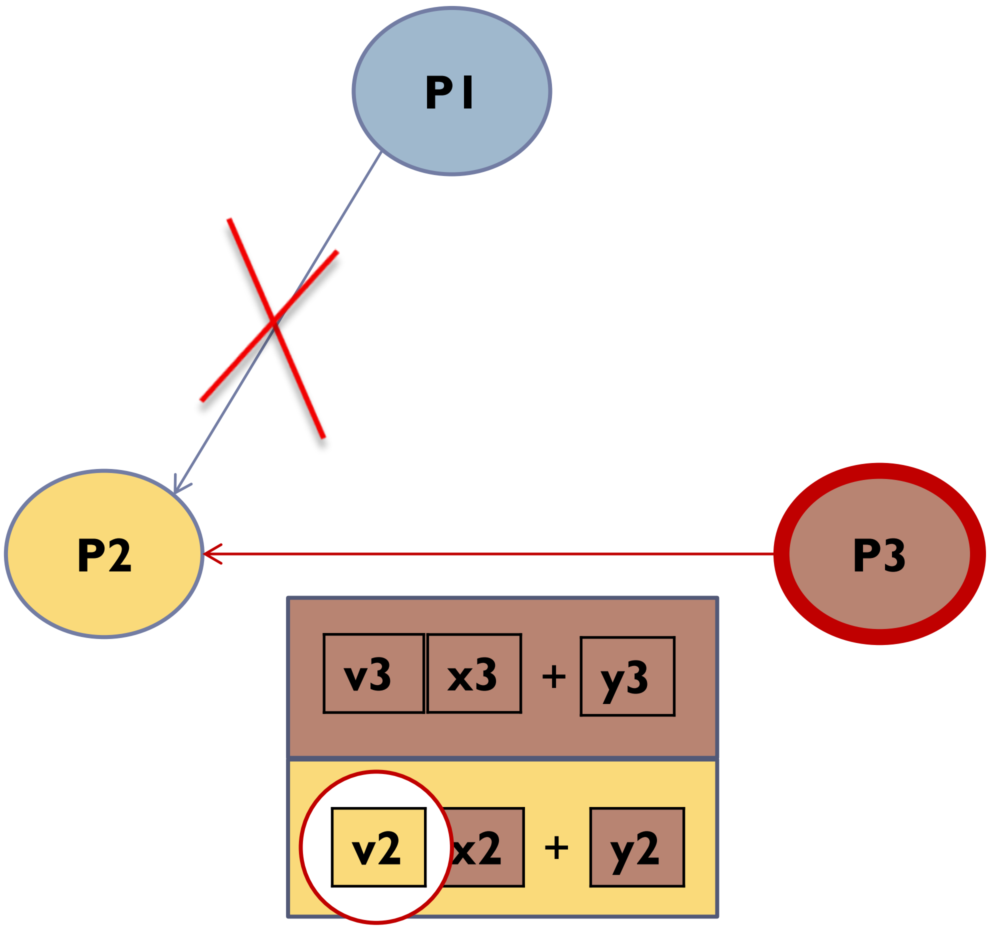
DSDP

- Normal case



DSDP: P3 is compromised

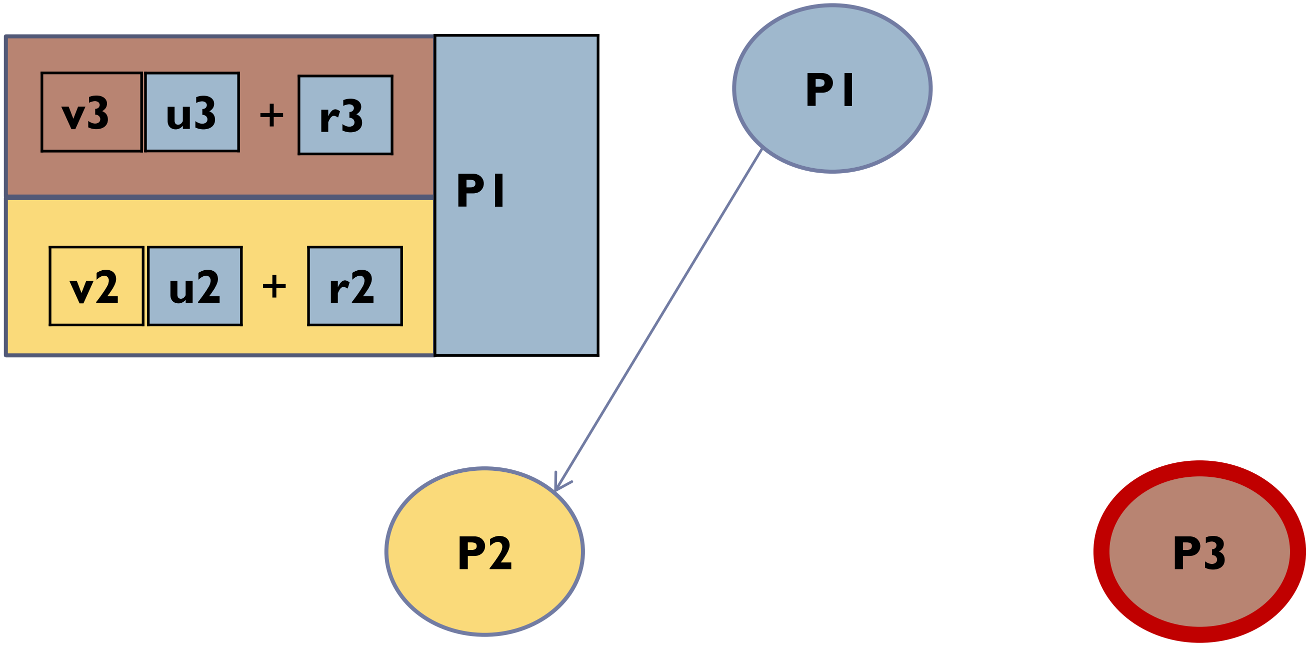
- Modified data sent from P3 instead of P1



DSDP: P3 is compromised

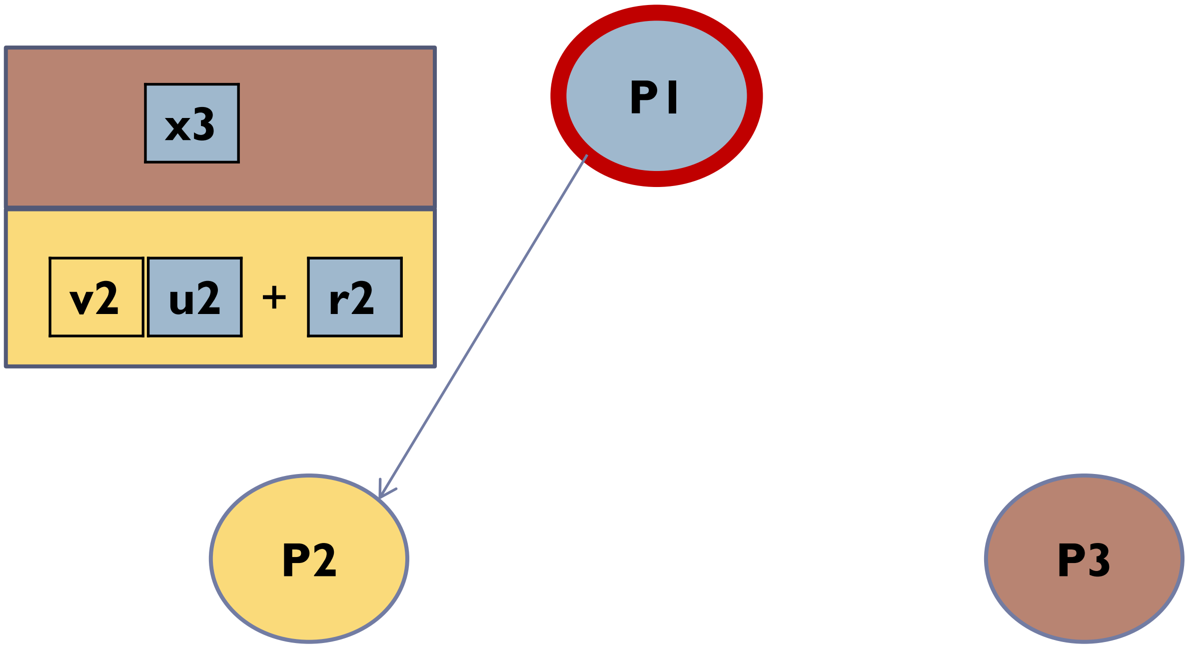
- Counter-measure:

  <u>Signatures</u>



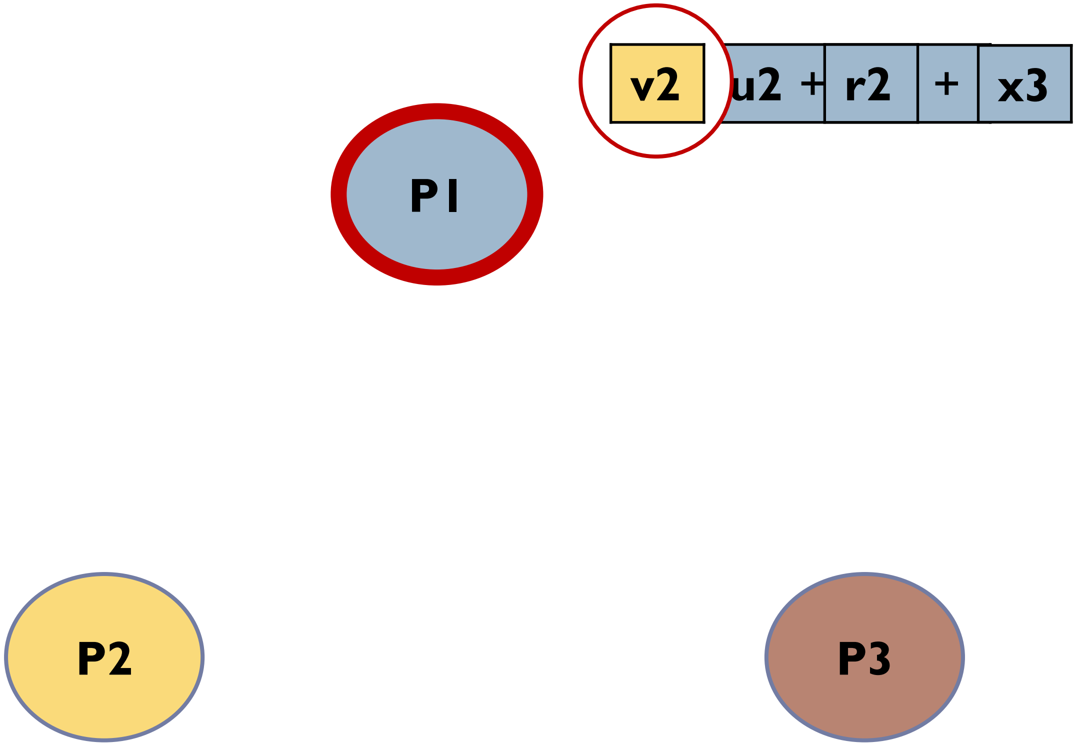
DSDP: P1 is compromised

- Attack: replacing u3 and r3



DSDP: P1 is compromised

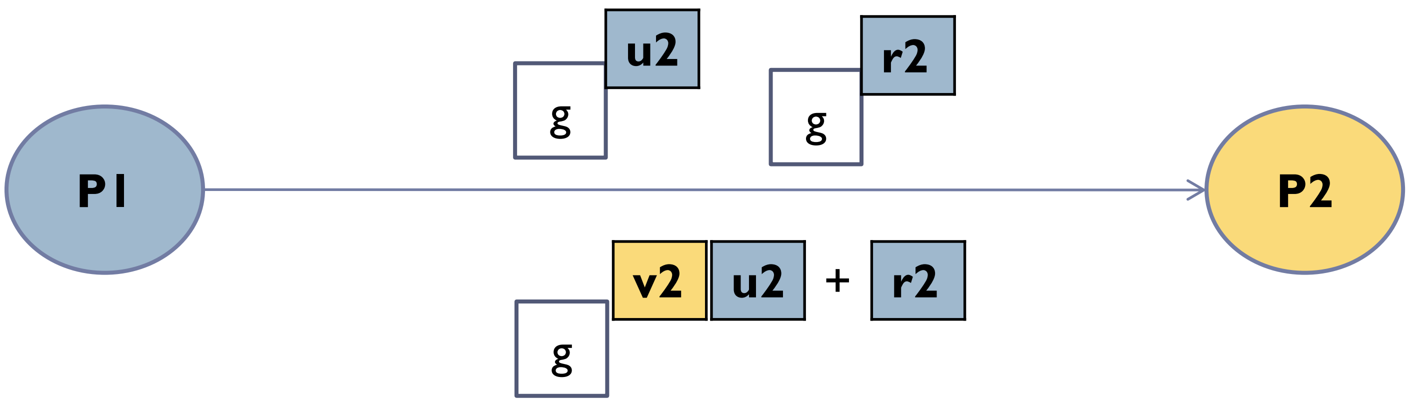
⇒ Only v2 is unknown!



DSDP: Counter-measure

- Zero-Knowledge

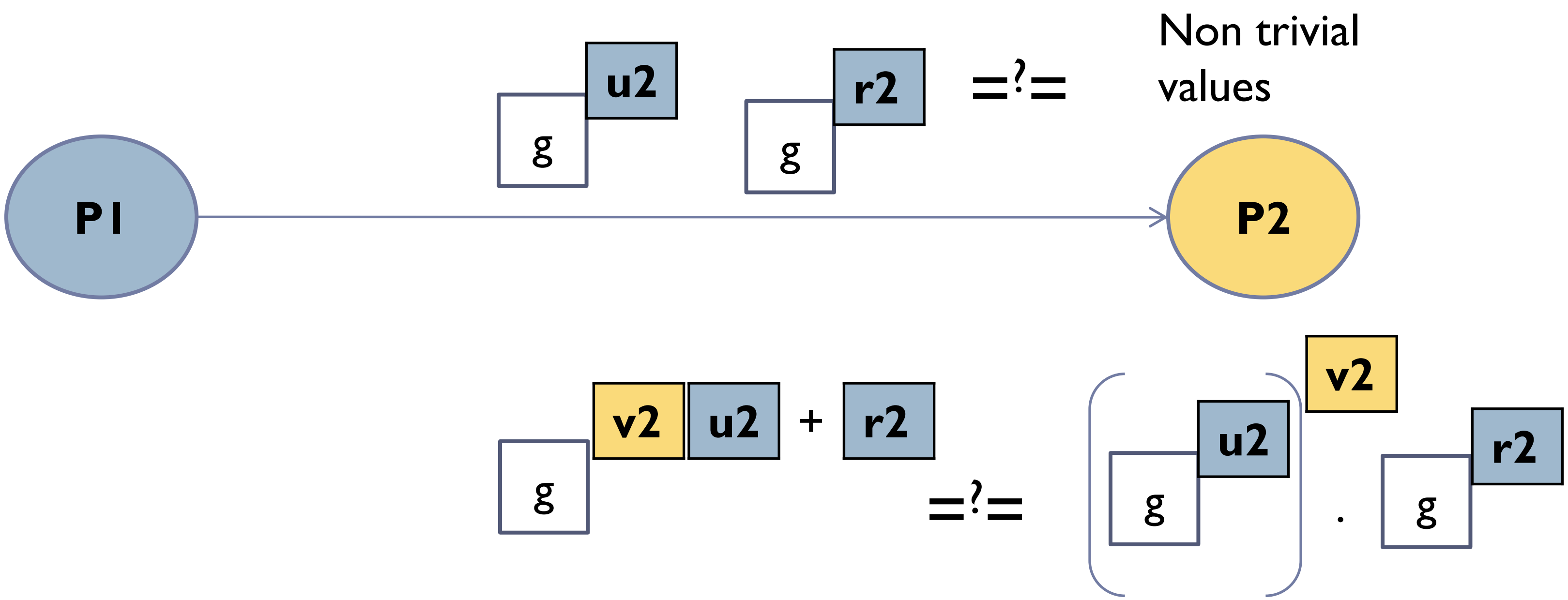
  Proof of non trivial affine transform



DSDP: Counter-measure

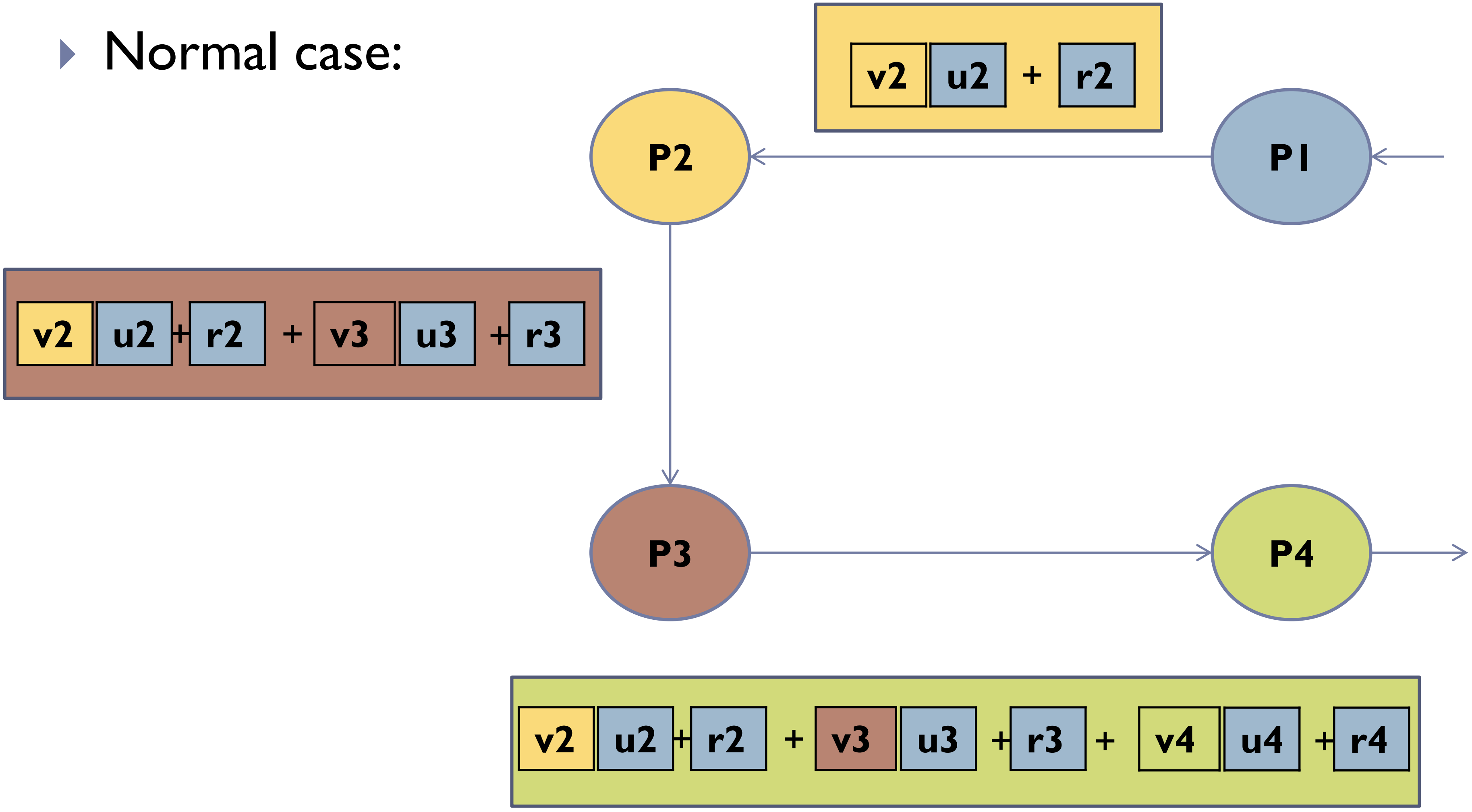
- <u>Zero-Knowledge</u>

  <u>Proof of non trivial affine transform</u>



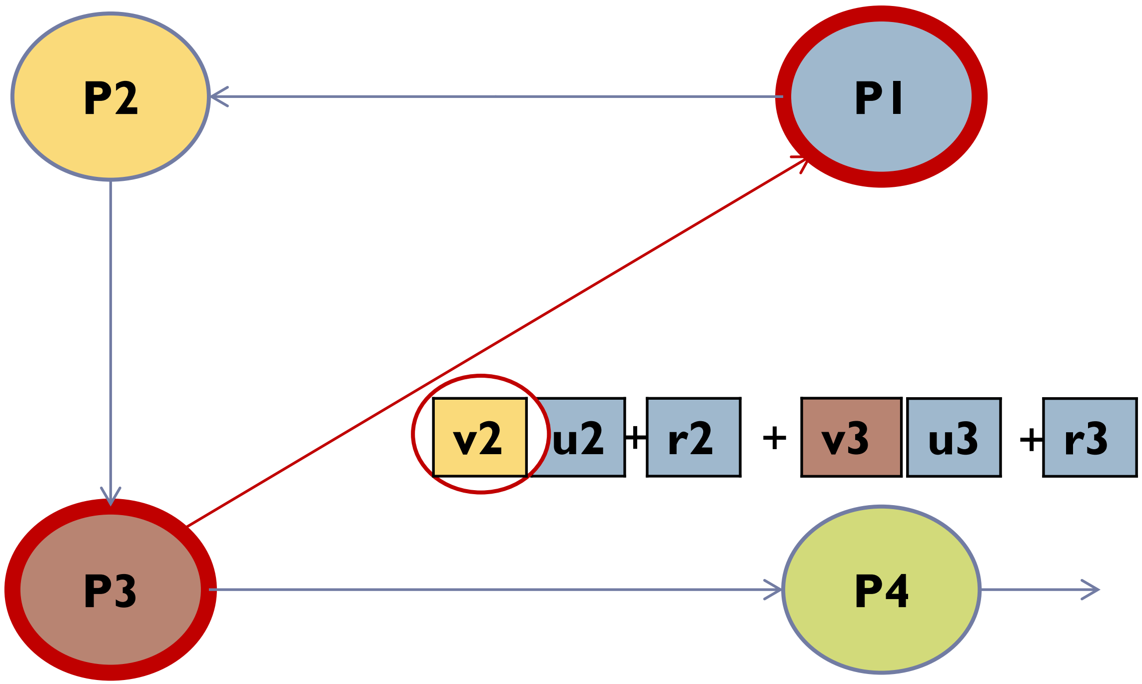
DSDP: Collusion Attack 1

- Normal case:



DSDP: P1 and P3 corrupted

- P3 extra data exchange:



DSDP: Collusion Attacks

- Attacks conditions:
  - P1 corrupted
  - Honest player rounded by corrupted ones

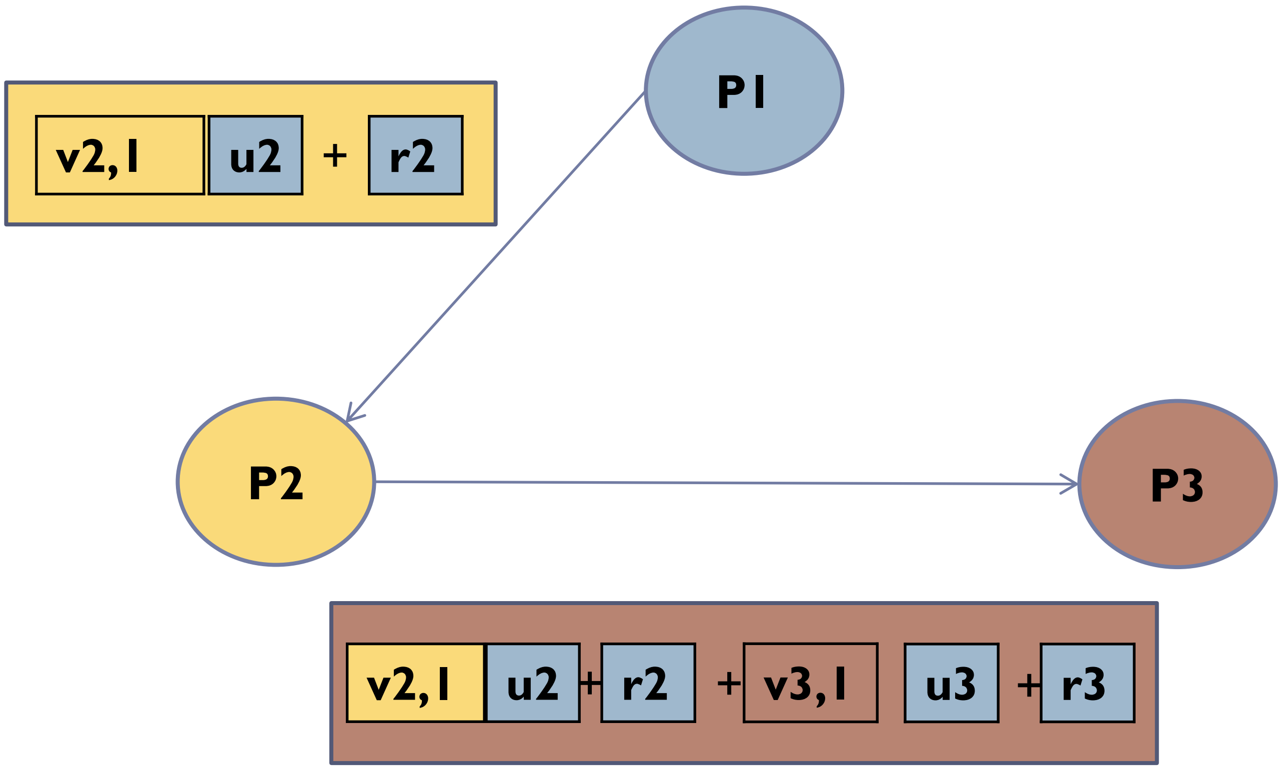
⇒ Problem: players' location!

- Counter-measure: Random Ring Order (RRO)
  - Players are randomly placed
  - *d* protocol repetitions

    using masked secrets

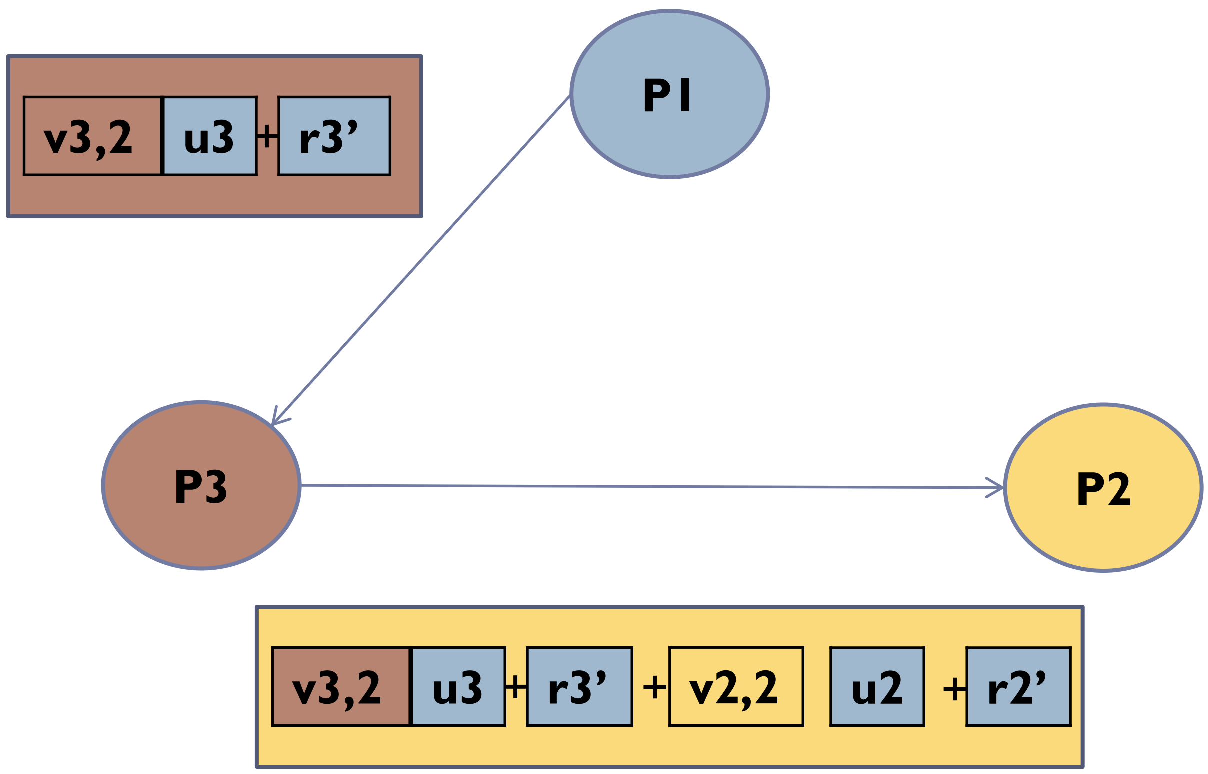
DSDP: Random Ring Order

- Masked secret: $v_i = v_{i,1} + v_{i,2}$
- Round 1:



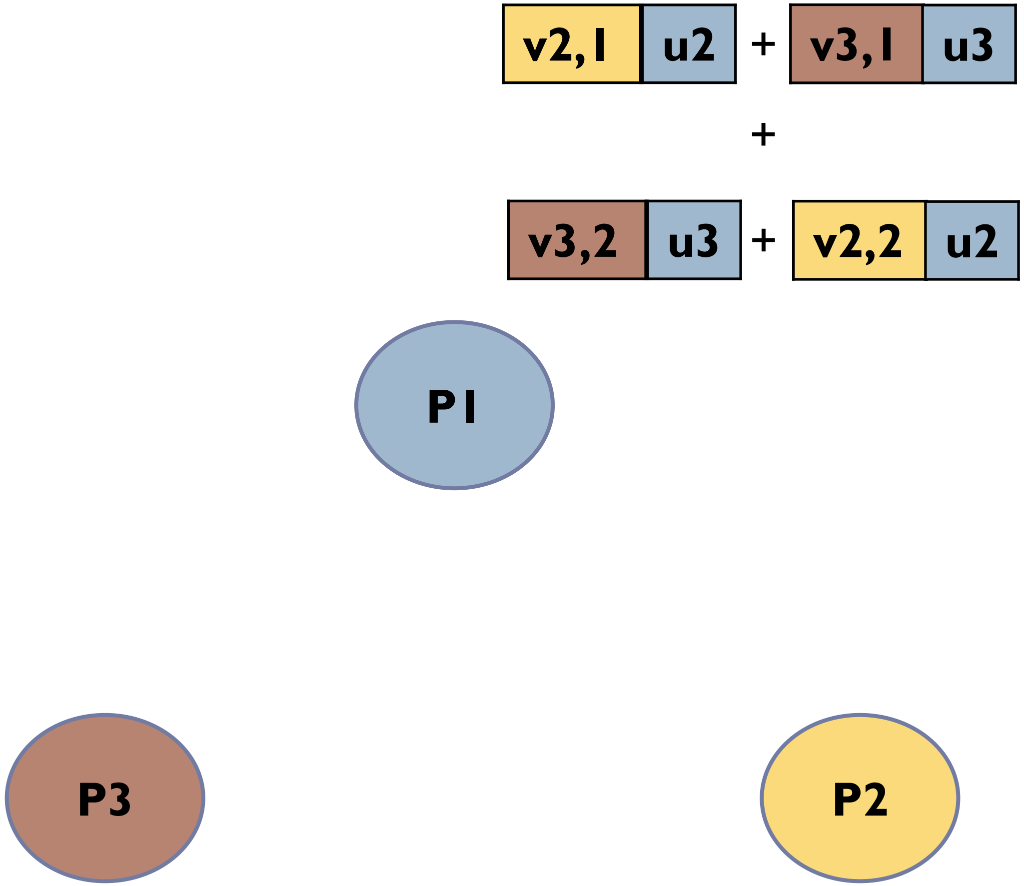
DSDP: Random Ring Order

- Masked secret: $v_i = v_{i,1} + v_{i,2}$
- <u>Round 2:</u>



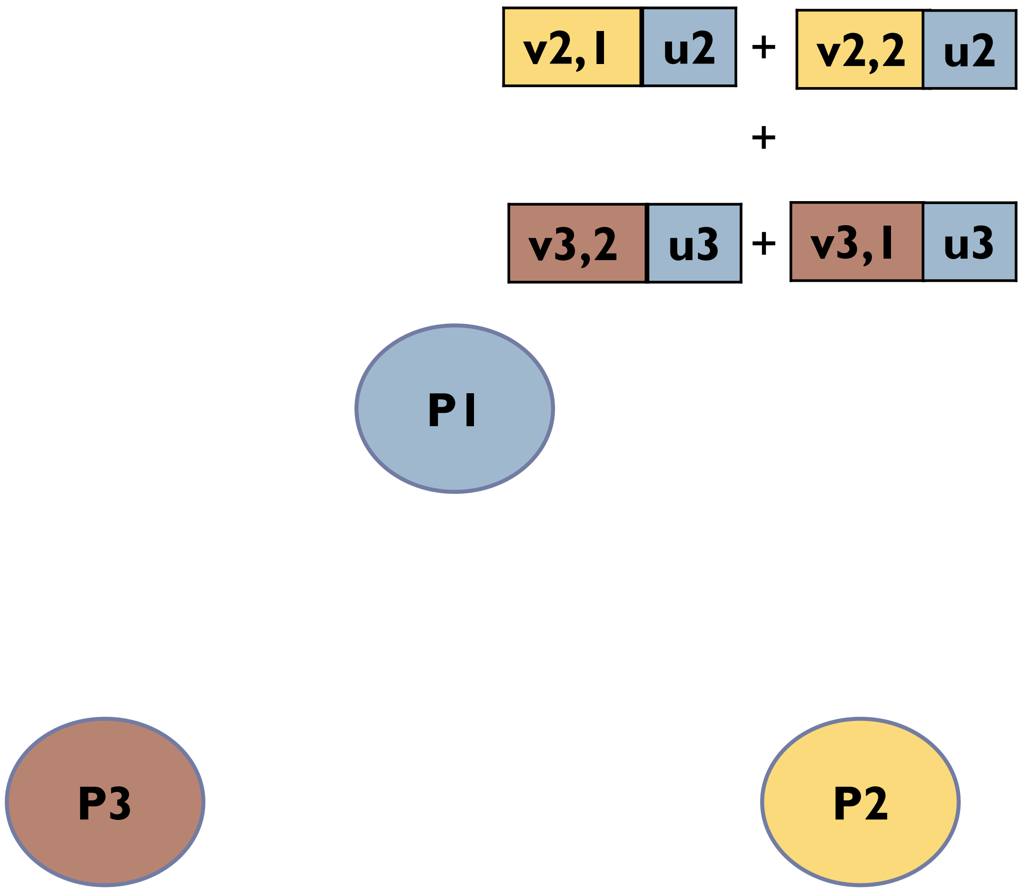
DSDP: Random Ring Order

- Masked secret: $v_i = v_{i,1} + v_{i,2}$
- <u>Last step:</u>



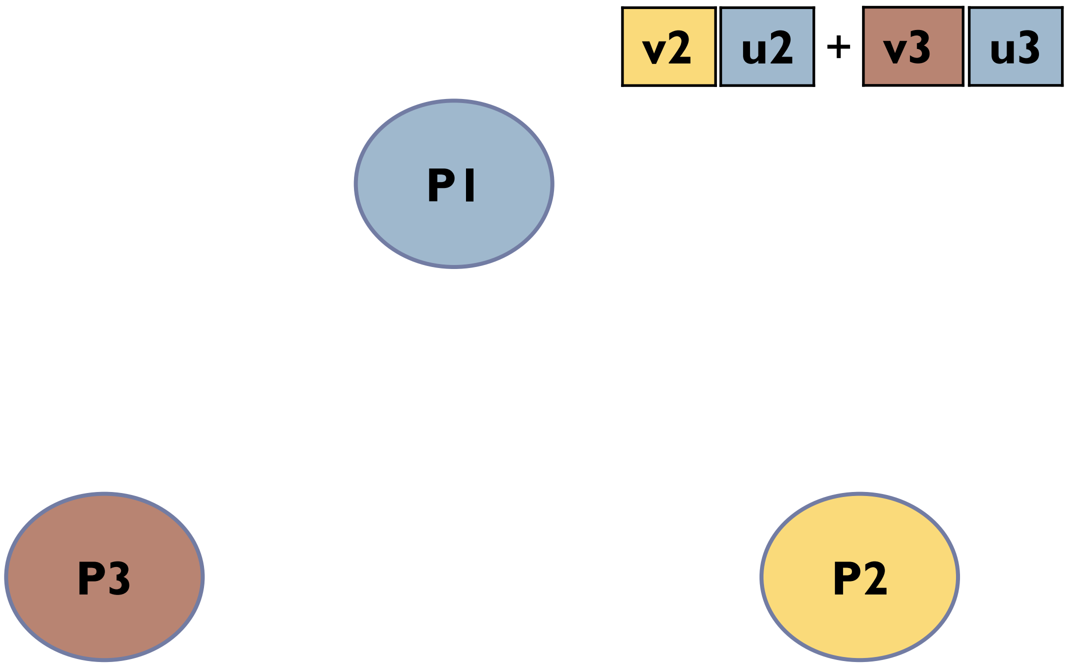
DSDP: Random Ring Order

- Masked secret: $v_i = v_{i,1} + v_{i,2}$
- <u>Last step:</u>



DSDP: Random Ring Order

- Masked secret: $v_i = v_{i,1} + v_{i,2}$
- <u>At the end:</u>



Security of RRO

- Attacks successful if:

  Adversaries are well-placed at each round

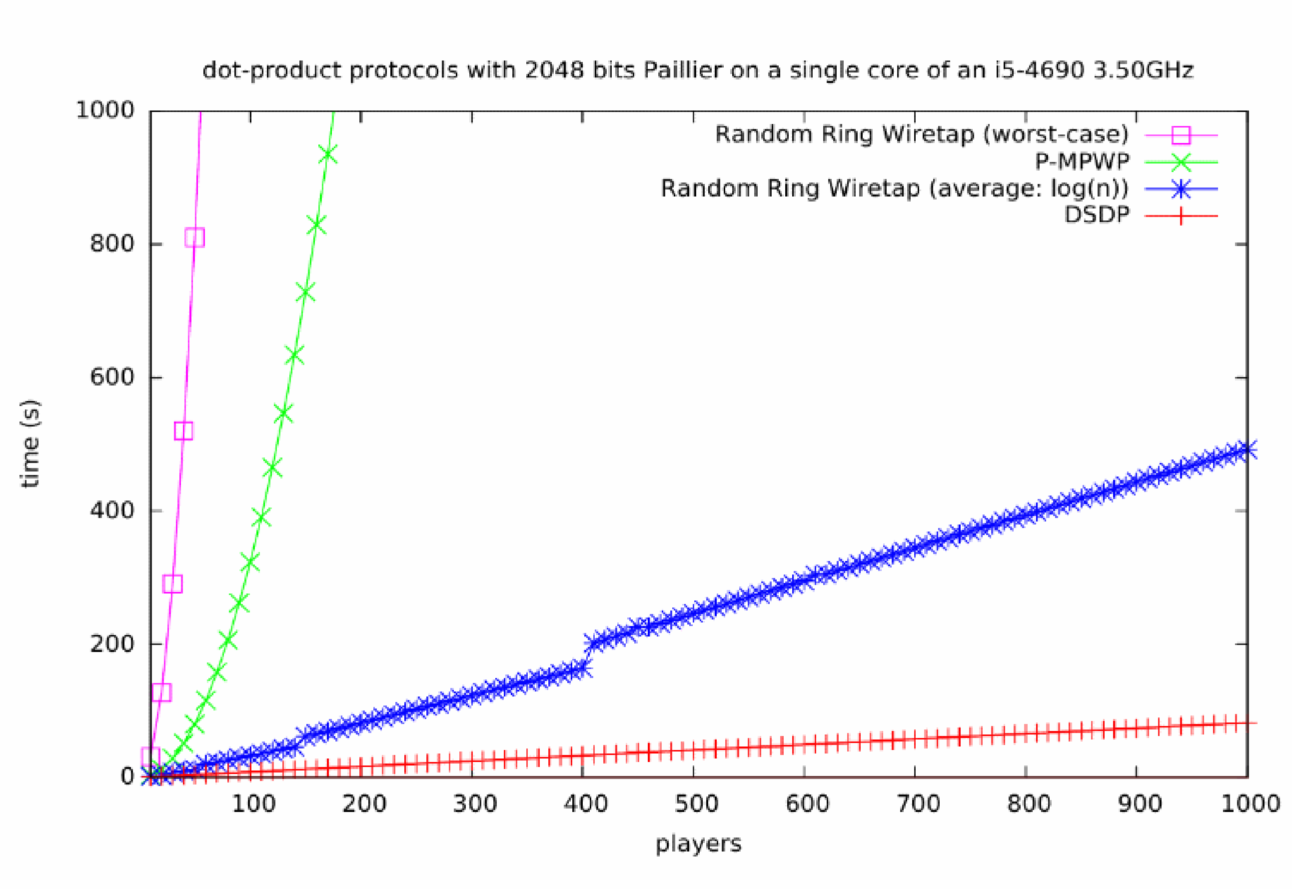
- Probabilist security:
  - $\#\{Malicious\ Players\} < \#\{Honests\ Players\}$

    => $d=O(\log n)$ rounds (in average)

- Guaranteed security:
  - Even in the worst case($\#\{Malicious\} = n-2$)

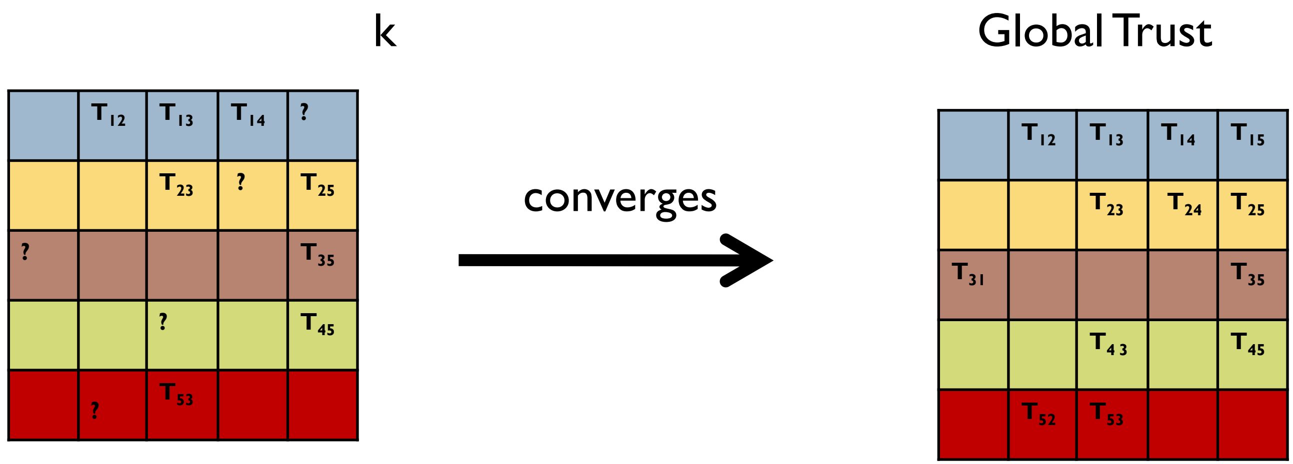
    => $d = O(n*\sigma)$ *rounds*, with σ bits of security

Dot Product Protocols Comparison



Private trust computation

- Applying dot-product protocols to matrix product



- Applicable to monoids of trust
- Inputs privacy

Conclusion

- Dot product protocols:
  - $O(n^2)$ secure against malicious adv.
  - $O(n)$ secure against honnest-but-curious adv.
  - $O(n\log(n))$ trade-off speed/security (RRO)
  - $O(n^2\sigma)$ to obtain guaranted security (RRO)
- From dot-product computations:
  - -> Matrix product
  - -> Trust computations
- Application:
  - Trust between certification authorities

Perspectives

- Comparison w/ a « dual » protocol

- Currently:
  - Paillier's cryptosystem

  ⇒ Efficiency with others cryptosystems ?

  (Naccache-Stern…)

- Matrix Multiplication:
  - DSDP: $O(n^3)$

⇒ Reducing to $O(n^\omega)$?

Thank you!

?