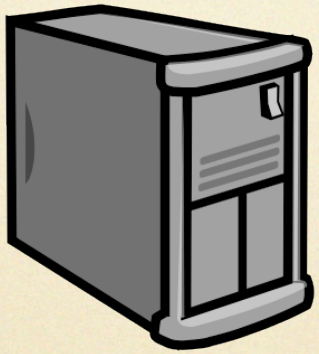
Securing services running over untrusted clouds : the two-tiered trust model

Aggelos Kiayias (U. Athens & U. Connecticut)

Joint work, **Juan Garay, Ran Gelles, David Johnson, Moti Yung**
(AT&T – UCLA - AT&T - Google)

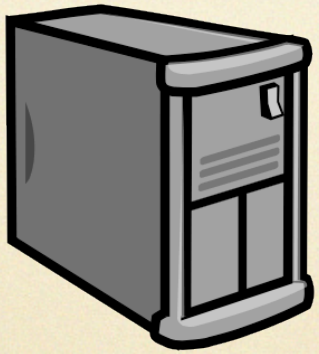
Outsourcing your service to the cloud

Moo(. . .)

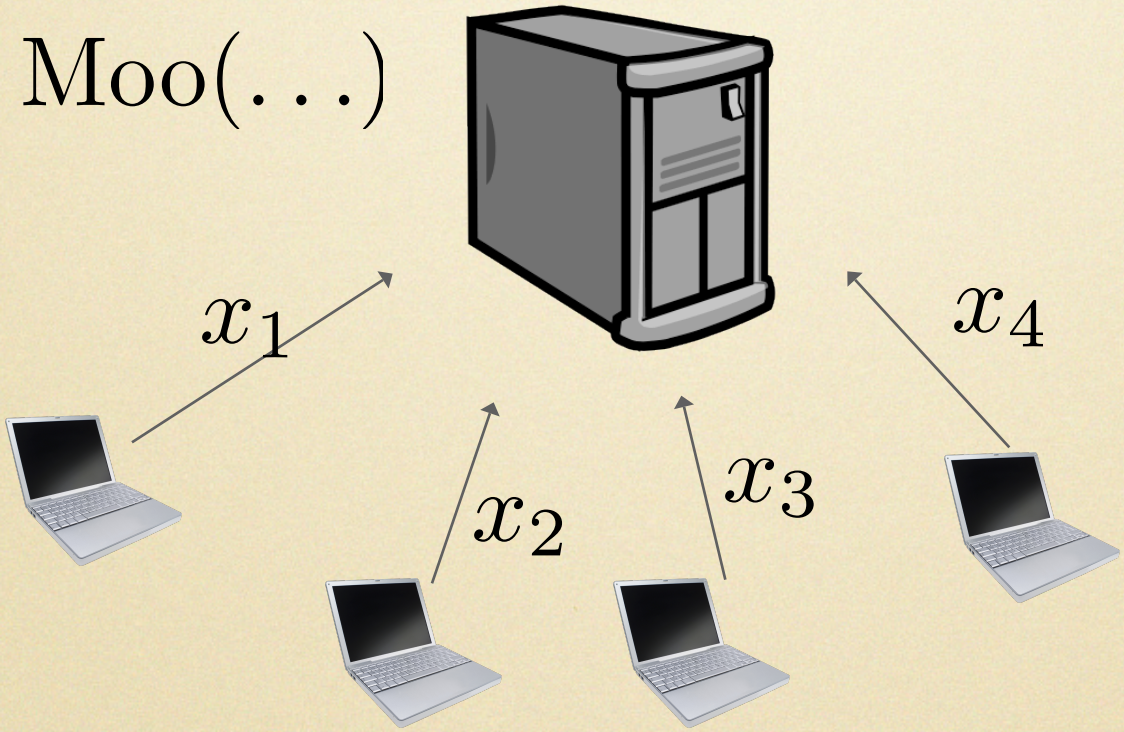


Outsourcing your service to the cloud

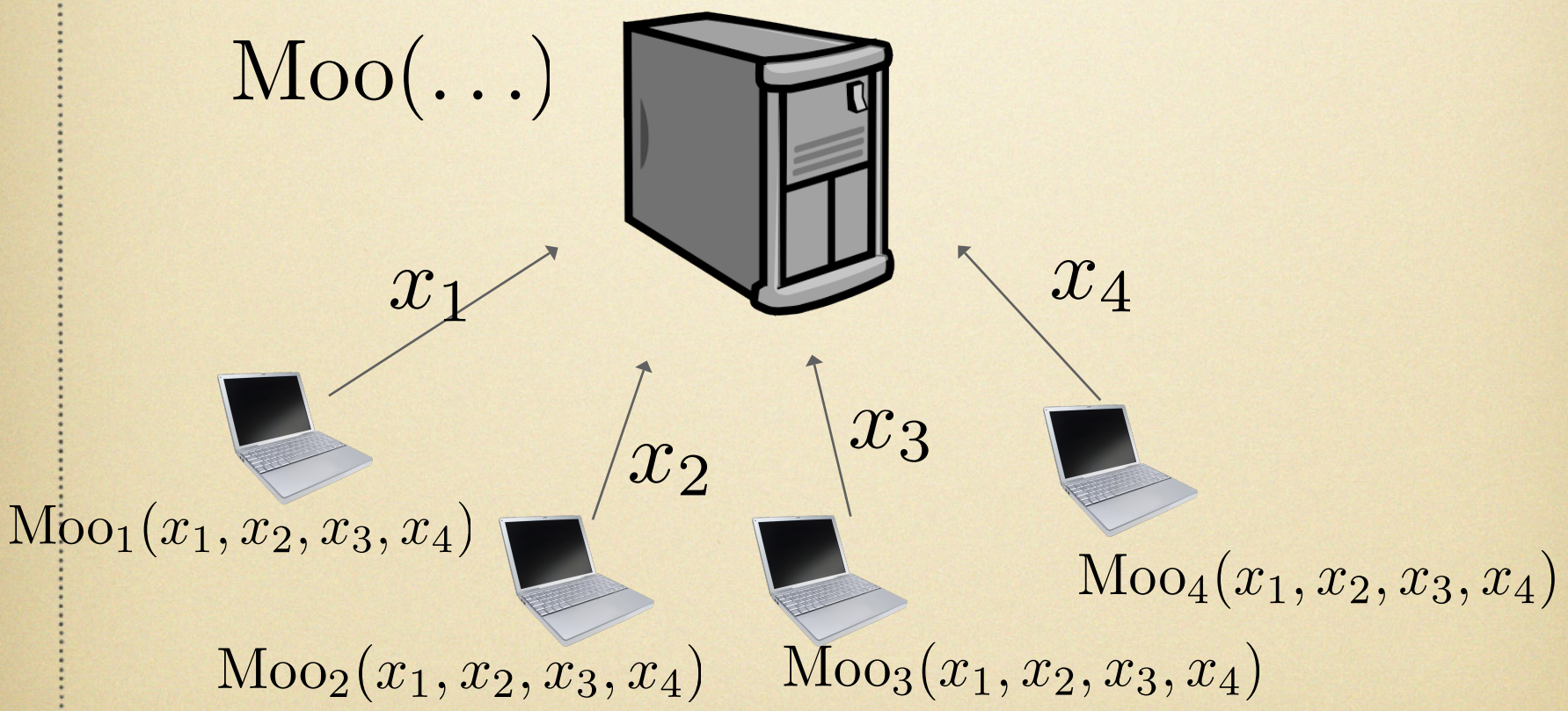
Moo(. . .)



Outsourcing your service to the cloud

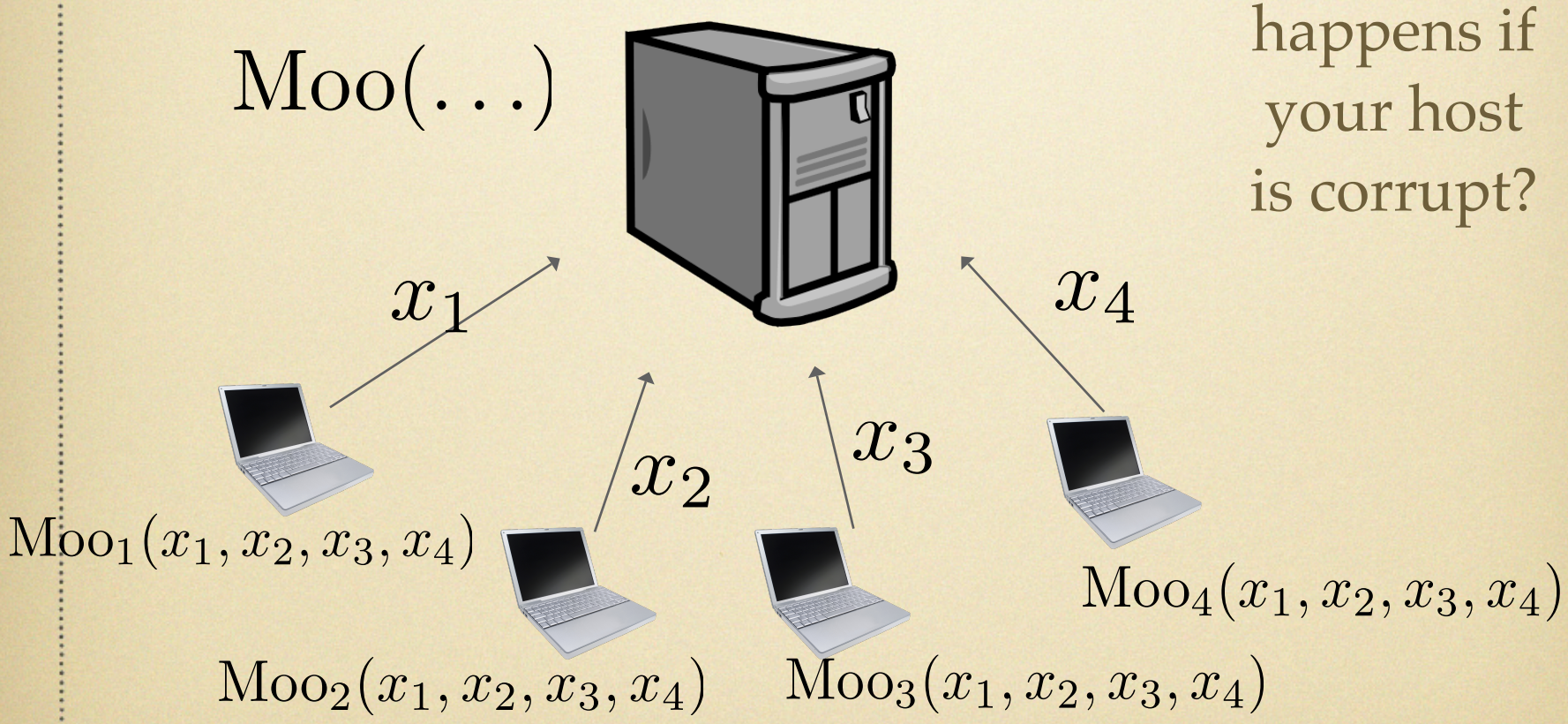


Outsourcing your service to the cloud



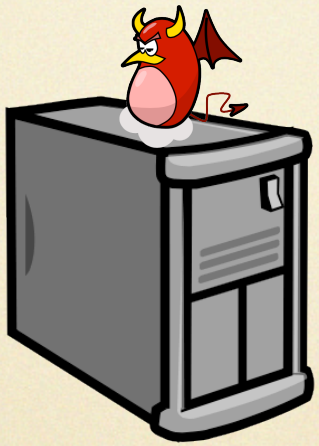
Outsourcing your service to the cloud

what happens if your host is corrupt?



Outsourcing your service to the cloud

what happens if your host is corrupt?



$\mathrm{Moo}(\ldots)$

x_1

x_2

x_3

x_4

$\mathrm{Moo}_1(x_1, x_2, x_3, x_4)$

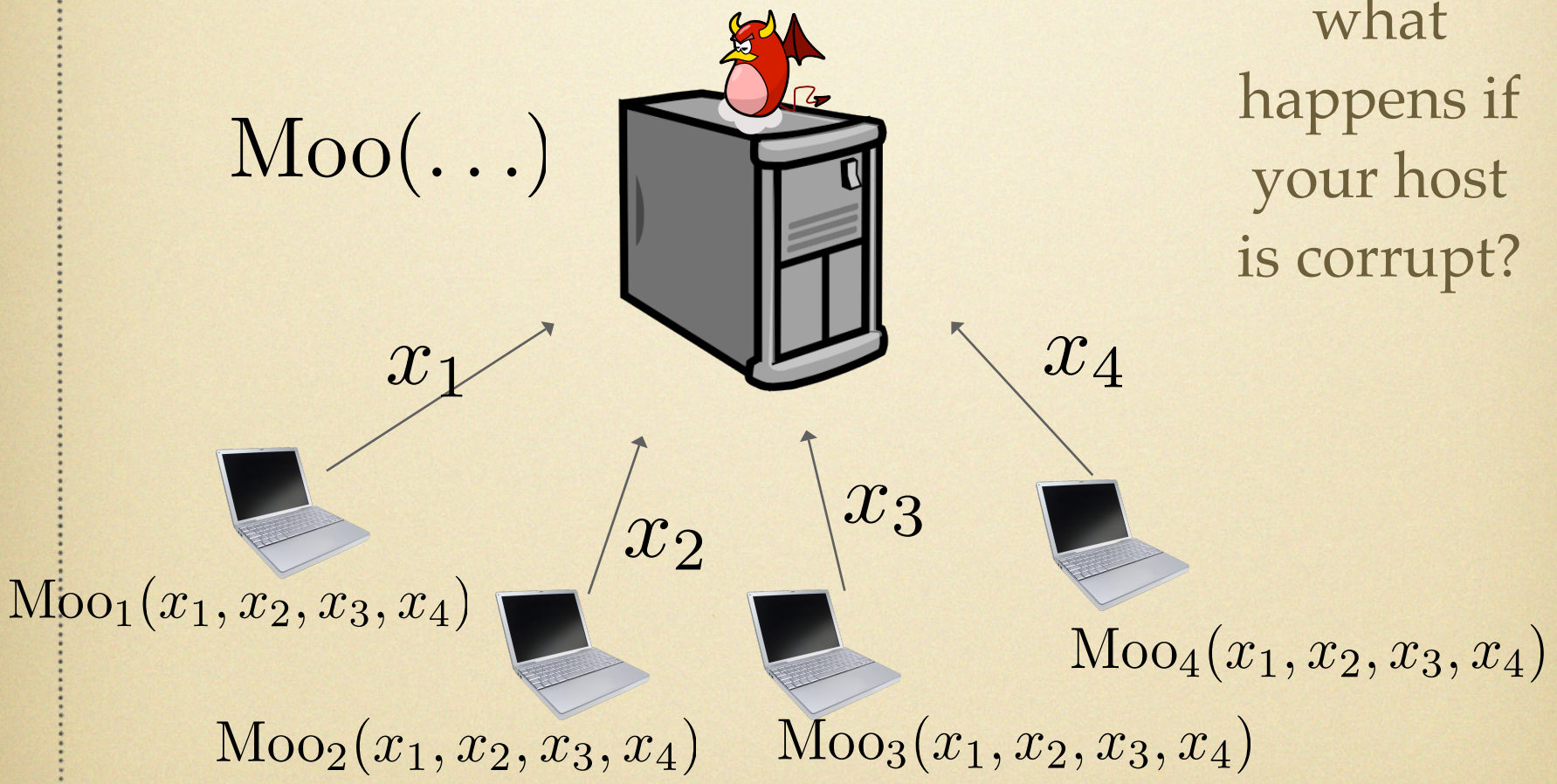
$\mathrm{Moo}_2(x_1, x_2, x_3, x_4)$

$\mathrm{Moo}_3(x_1, x_2, x_3, x_4)$

$\mathrm{Moo}_4(x_1, x_2, x_3, x_4)$

Outsourcing your service to the cloud

what happens if your host is corrupt?

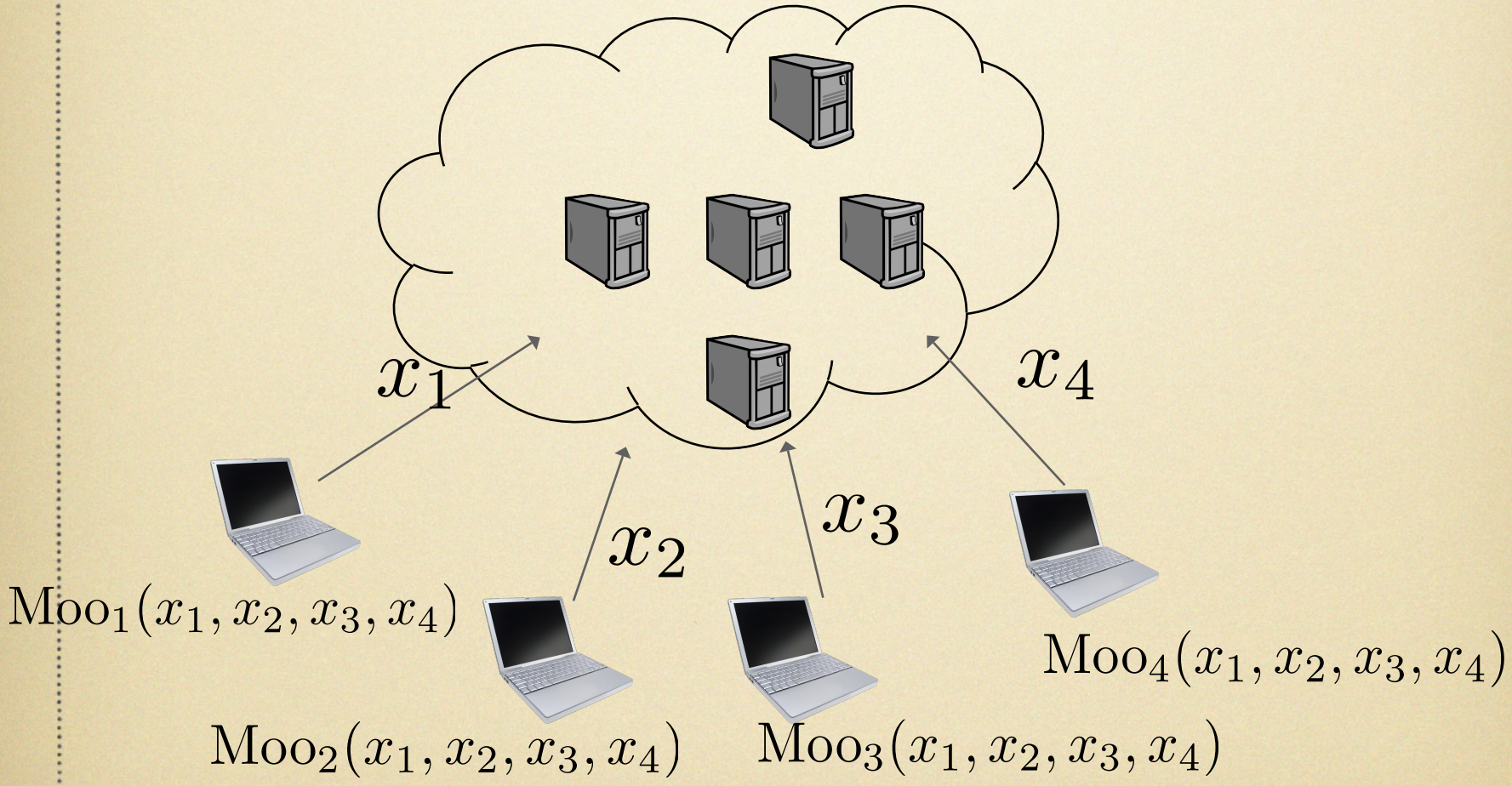


encryption / signatures cannot help here

what is at stake?

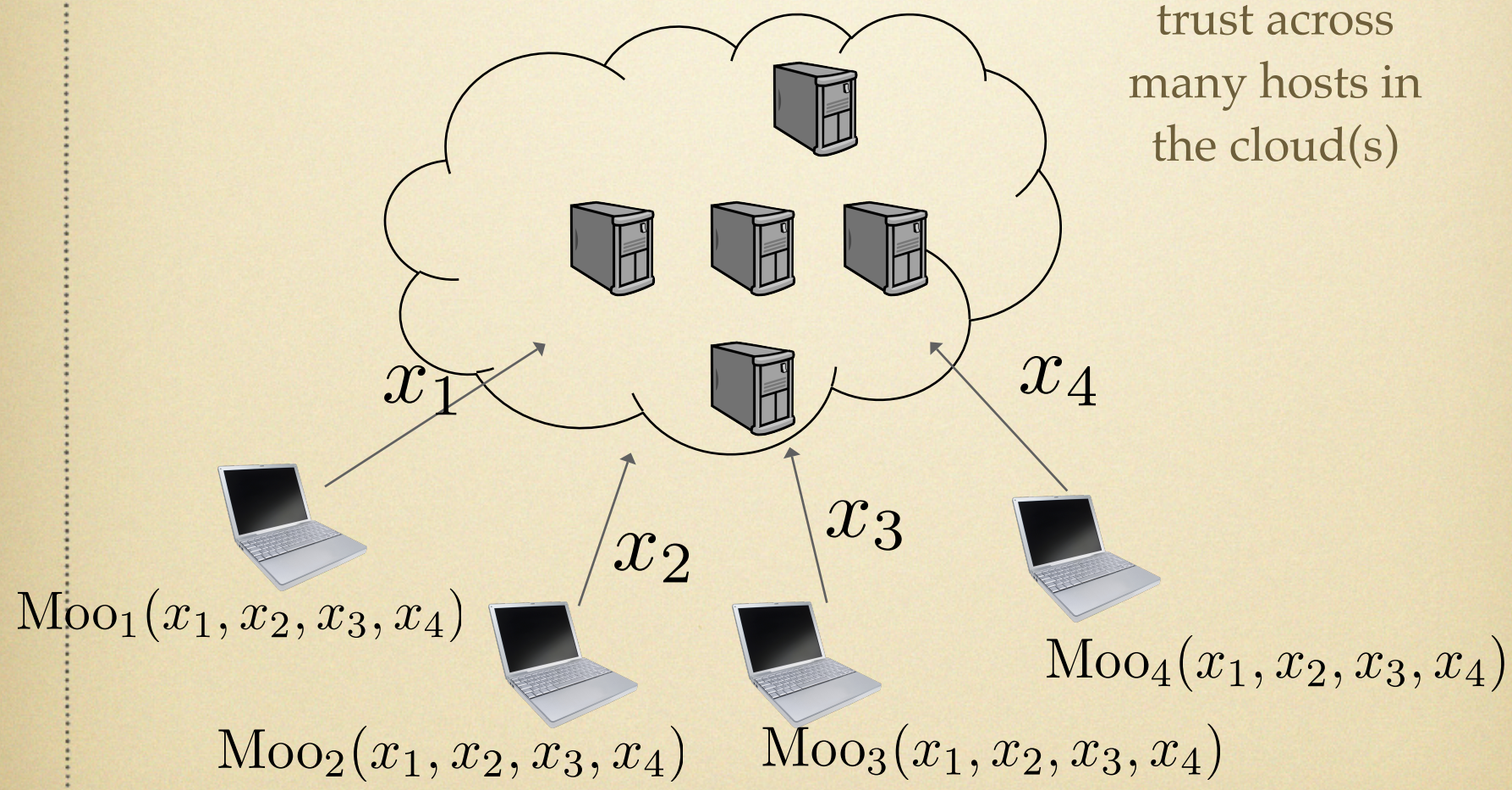
- Privacy of user inputs.
- Guaranteed Output Delivery.
- Fairness.
- Input-independence.

Distributing Trust



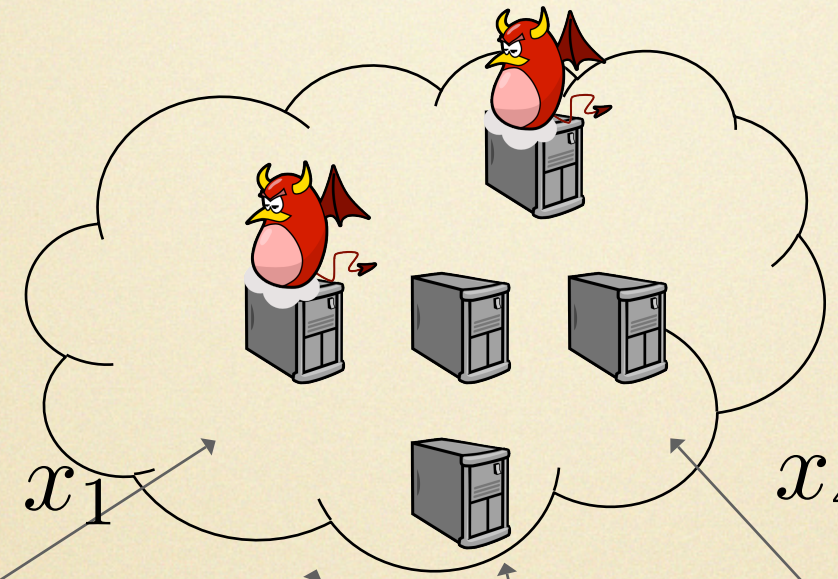
Distributing Trust

Spread trust across many hosts in the cloud(s)



Distributing Trust

Spread trust across many hosts in the cloud(s)



x_1

x_2

x_3

x_4

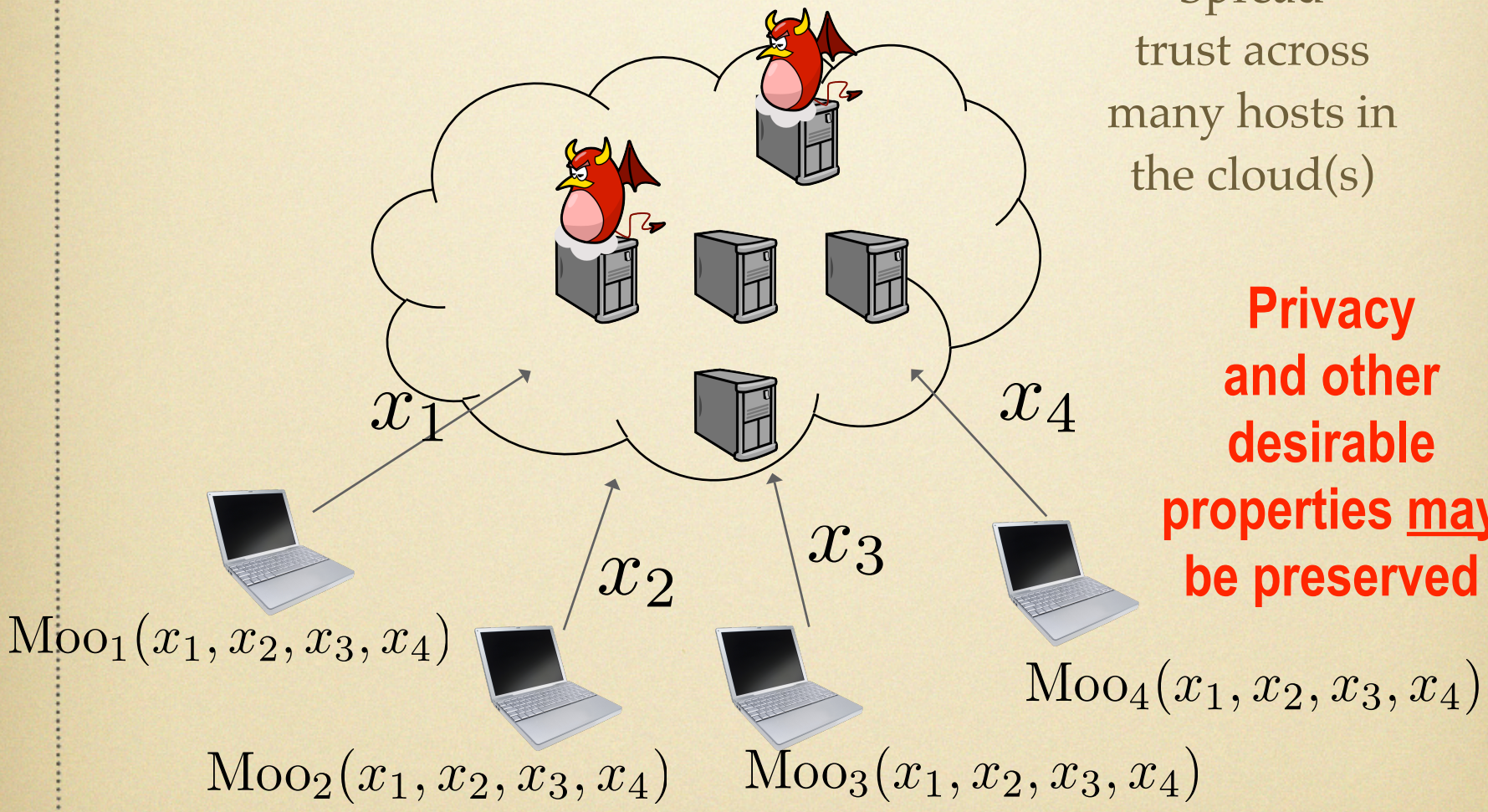
$\mathrm{Moo}_1(x_1, x_2, x_3, x_4)$

$\mathrm{Moo}_2(x_1, x_2, x_3, x_4)$

$\mathrm{Moo}_3(x_1, x_2, x_3, x_4)$

$\mathrm{Moo}_4(x_1, x_2, x_3, x_4)$

Distributing Trust



Spread trust across many hosts in the cloud(s)

Privacy and other desirable properties may be preserved

Crypto Protocols

for distributing trust.

Crypto Protocols

for distributing trust.

- An adversary controlling any *minority* of the servers cannot prevent the secure computation of *any efficient functionality* defined over their inputs [Yao82, GMW87]
- Similar results hold over secure channels (and no add'l crypto) with an (computationally unbounded) adversary controlling less than a *third* of the servers [BGW88, CCD88]

Crypto Protocols

for distributing trust.

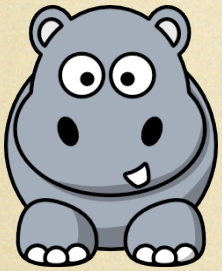
- An adversary controlling any *minority* of the servers cannot prevent the secure computation of *any efficient functionality* defined over their inputs [Yao82, GMW87]
- Similar results hold over secure channels (and no add'l crypto) with an (computationally unbounded) adversary controlling less than a *third* of the servers [BGW88, CCD88]

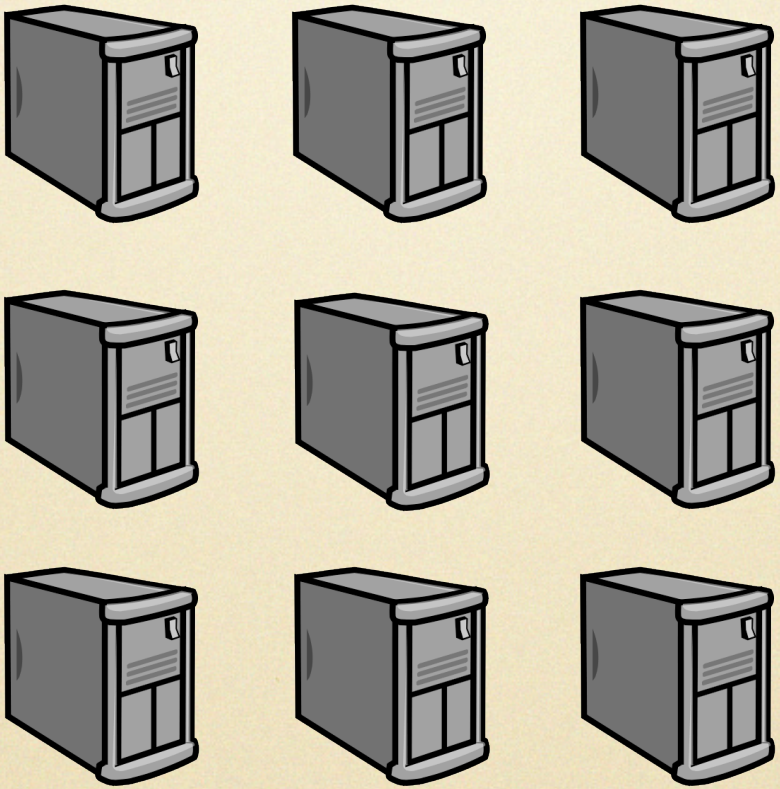
Crypto Protocols

for distributing trust.

- An adversary controlling any *minority* of the servers cannot prevent the secure computation of *any efficient functionality* defined over their inputs [Yao82, GMW87]
- Similar results hold over secure channels (and no add'l crypto) with an (computationally unbounded) adversary controlling less than a *third* of the servers [BGW88, CCD88]

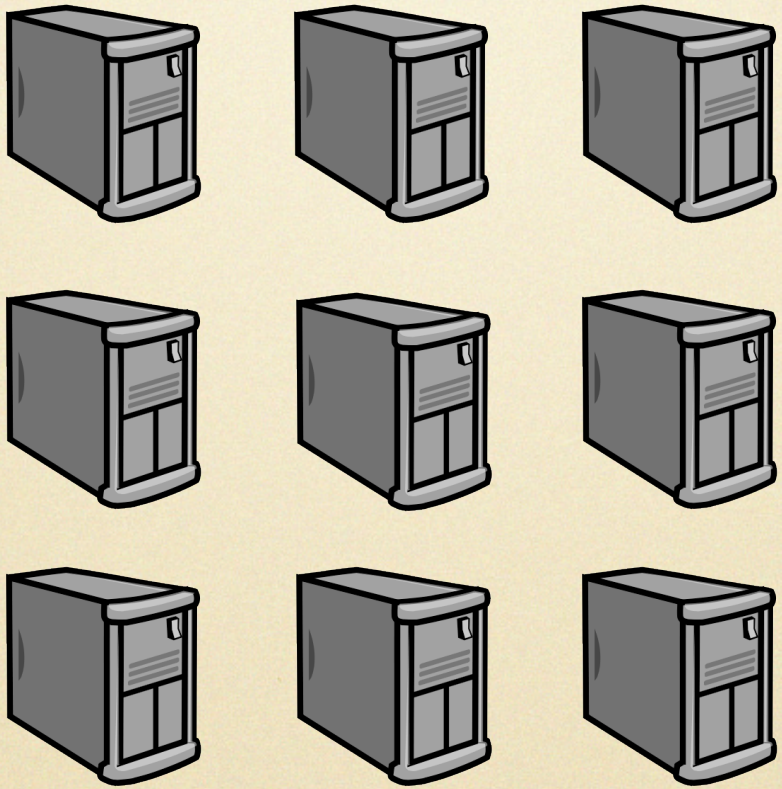
the doomsday scenario





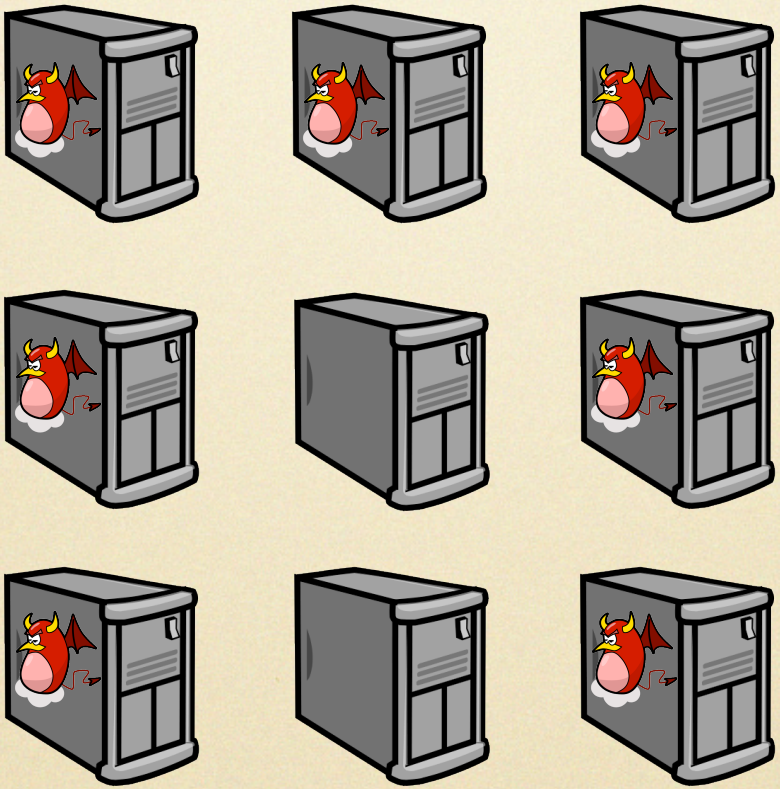
The service provider may never be entirely sure

the doomsday scenario



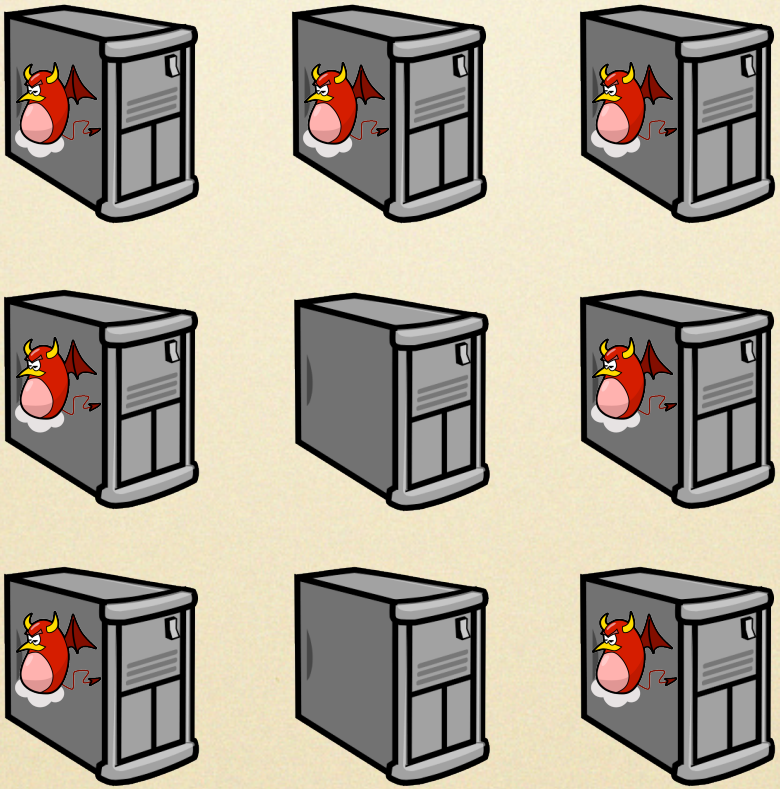
The service provider may never be entirely sure

the doomsday scenario



The service provider may never be entirely sure

the doomsday scenario

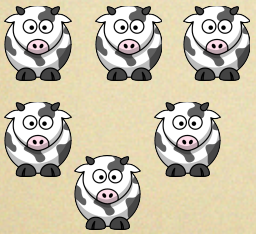


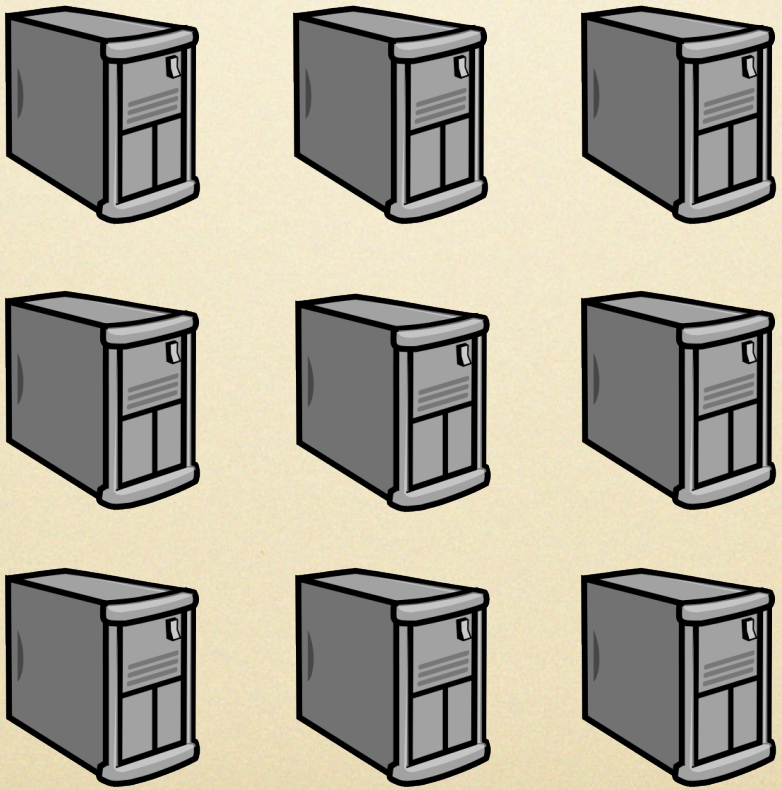
The service provider may never be entirely sure

the service may be subverted from the get-go

the outsourcing SP point of view

the SP is considering to start an MPC service over a cloud of servers



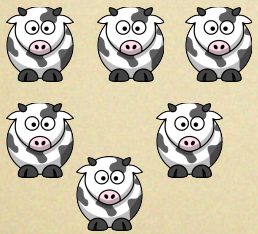


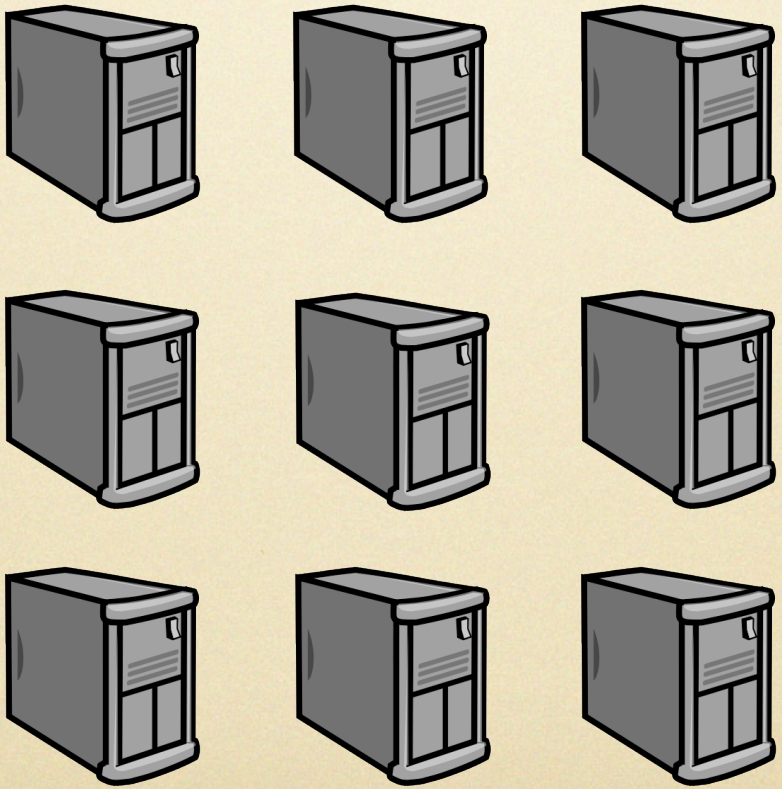
The cloud is already operational

what is the trust guarantee it provides?

the outsourcing SP point of view

the SP is considering to start an MPC service over a cloud of servers



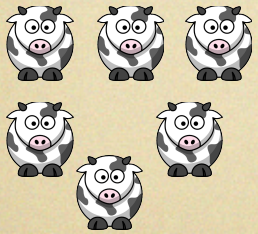


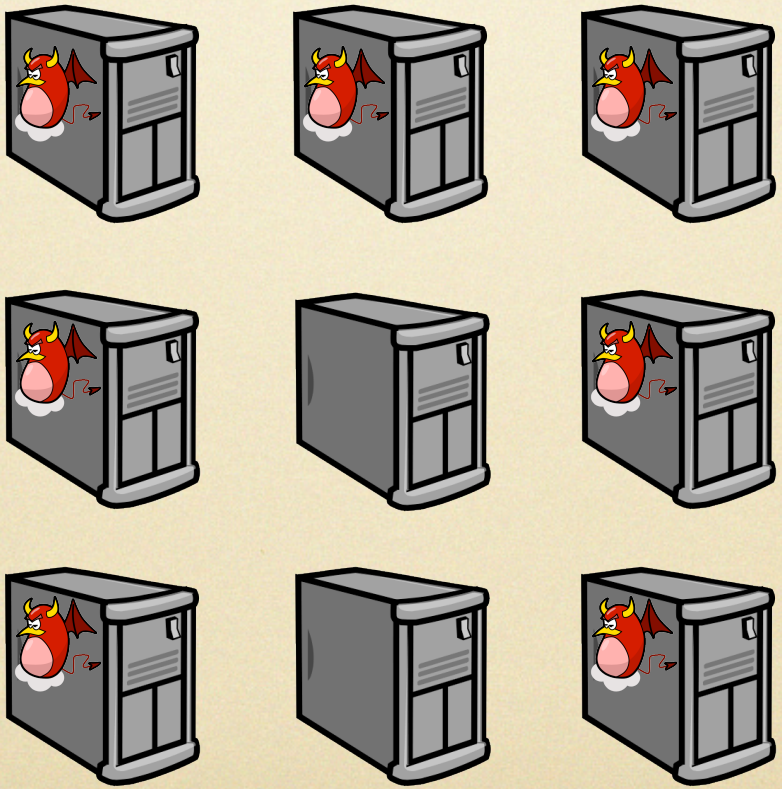
The cloud is already operational

what is the trust guarantee it provides?

the outsourcing SP point of view

the SP is considering to start an MPC service over a cloud of servers





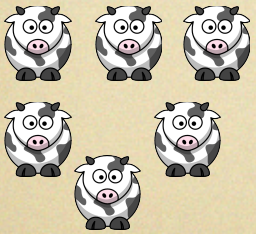
The cloud is already operational

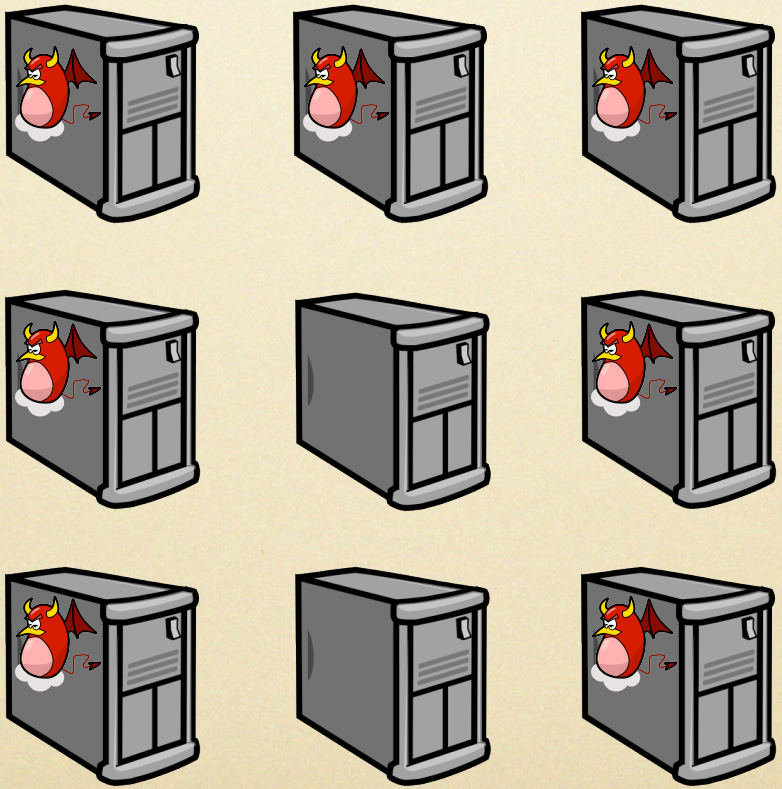
what is the trust guarantee it provides?

the outsourcing SP point of view

the SP is considering to start an MPC service over a cloud of servers

?



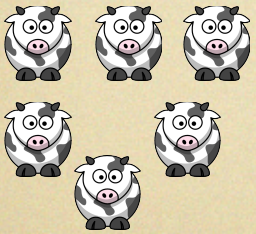


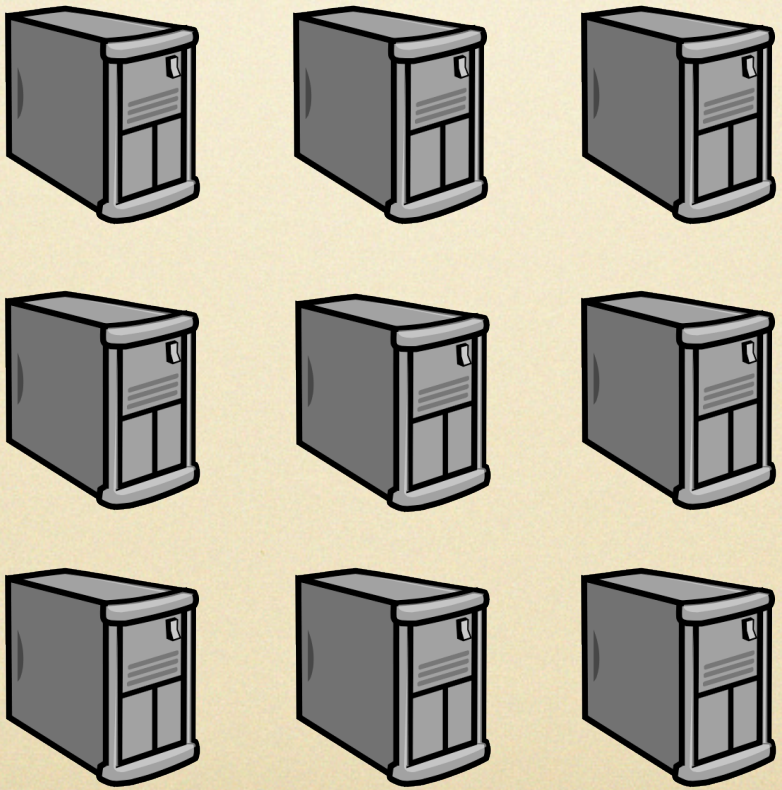
The cloud is already operational

what is the trust guarantee it provides?

the outsourcing SP point of view

the SP is considering to start an MPC service over a cloud of servers





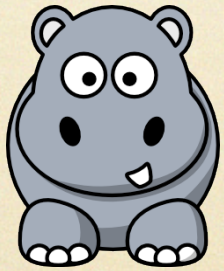
The cloud is already operational

what is the trust guarantee it provides?

the outsourcing SP point of view, 2

how certain are you about your the hosts provided by your cloud provider(s) ?

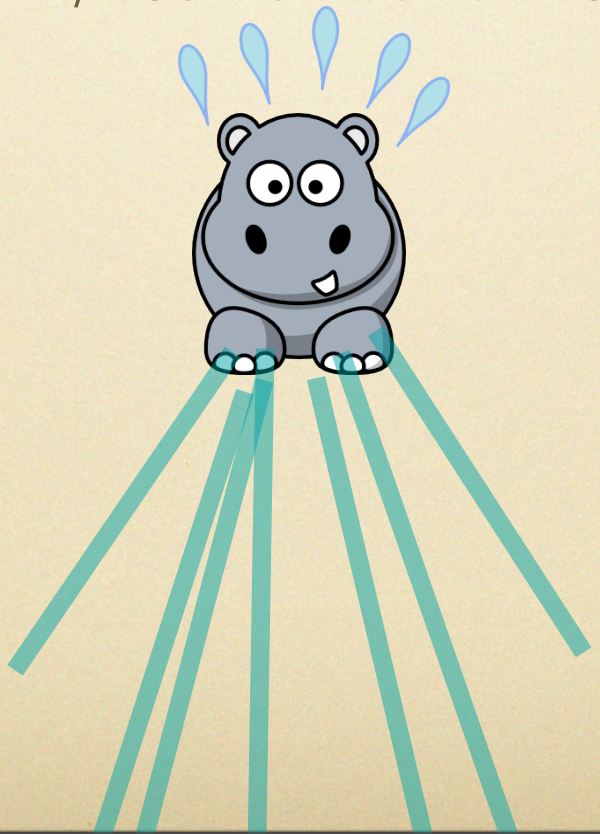
how much effort/cost is needed to become certain?



the outsourcing SP point of view, 2

how certain are you about your the hosts provided by your cloud provider(s) ?

how much effort/cost is needed to become certain?



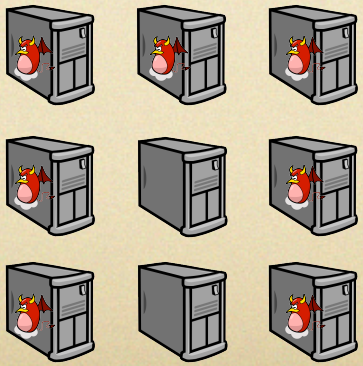
abstracting it as a game : SP vs adversary

1. Introduces n servers

2. Corrupts an α fraction of servers

3. Decides to initiate an MPC service. Inspects a fraction β of servers & possibly repairs

4. After service commences continues to corrupt a γ fraction

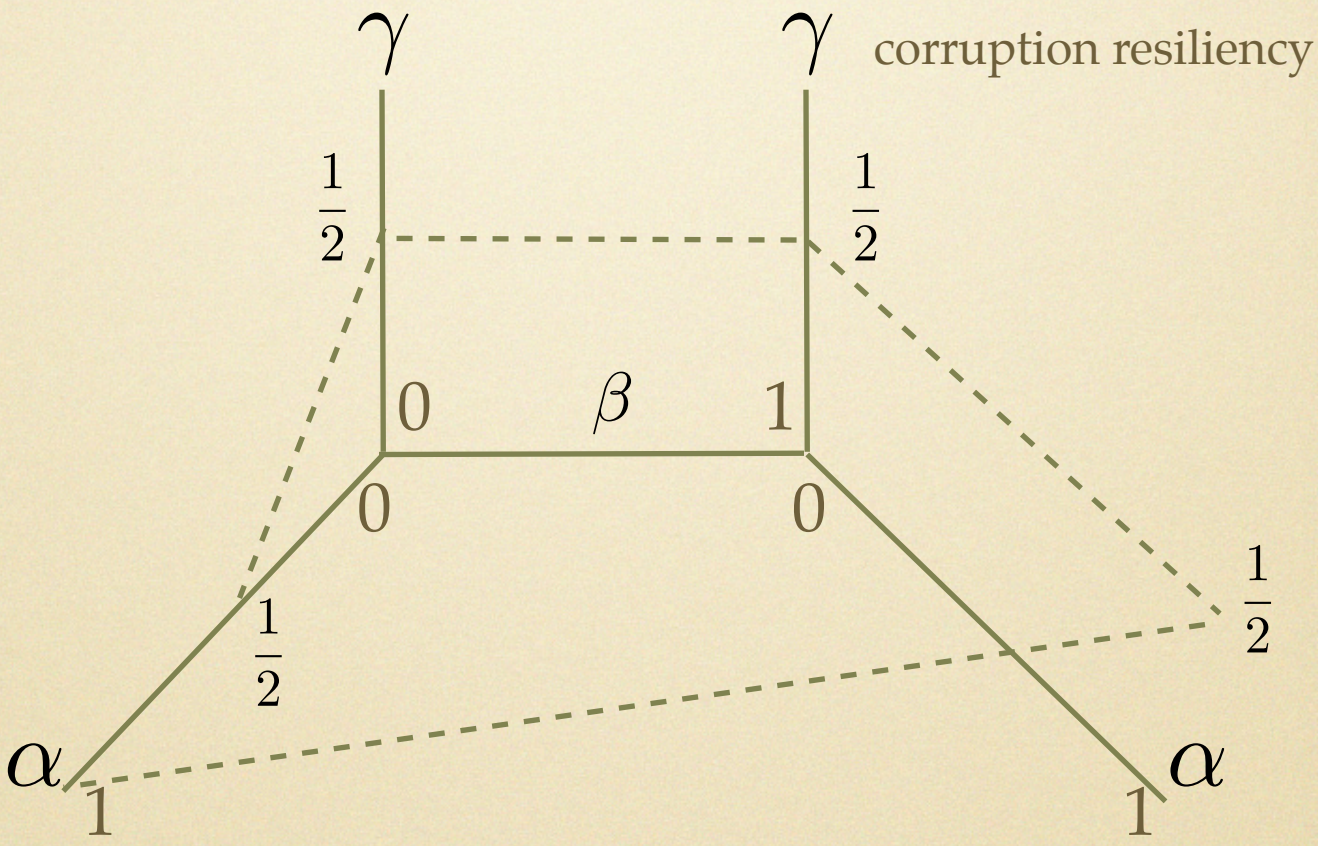


for what values of (α, β, γ) the service can be maintained?

to simplify: assume (α, β, γ) public

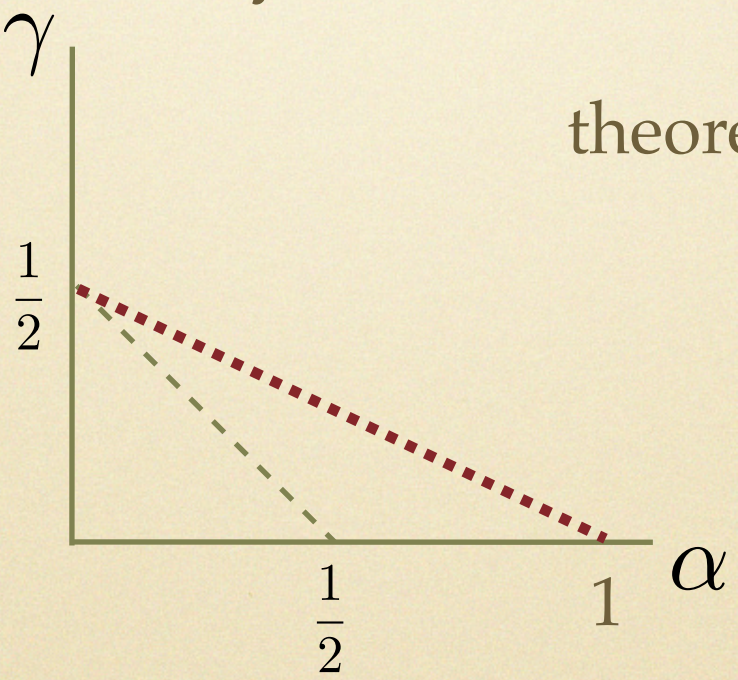
The boundary cases

are implied by standard cryptographic results



The $\beta = 0$ case.

corruption resiliency



theoretically available:

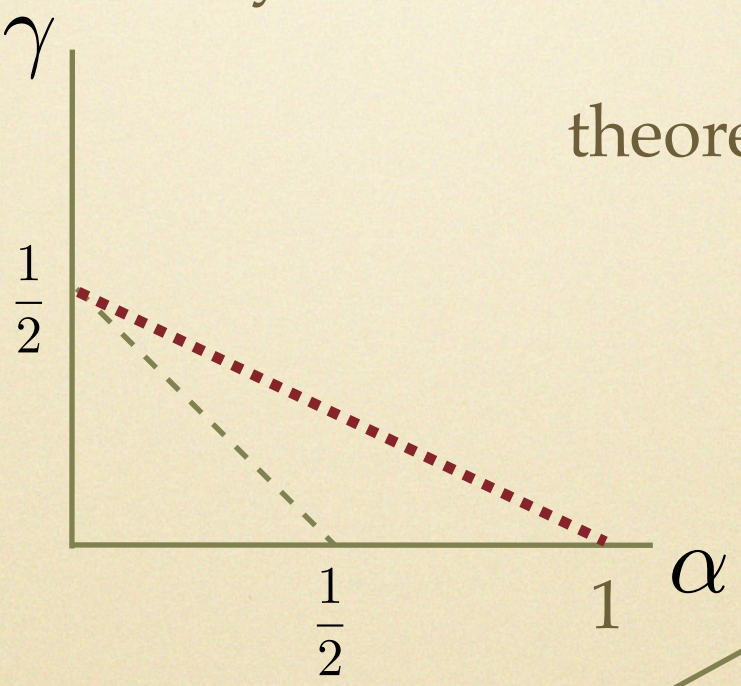
$$\gamma = \frac{1-\alpha}{2}$$

achievable via standard crypto:

$$\gamma = \frac{1}{2} - \alpha$$

The $\beta = 0$ case.

corruption resiliency



theoretically available:

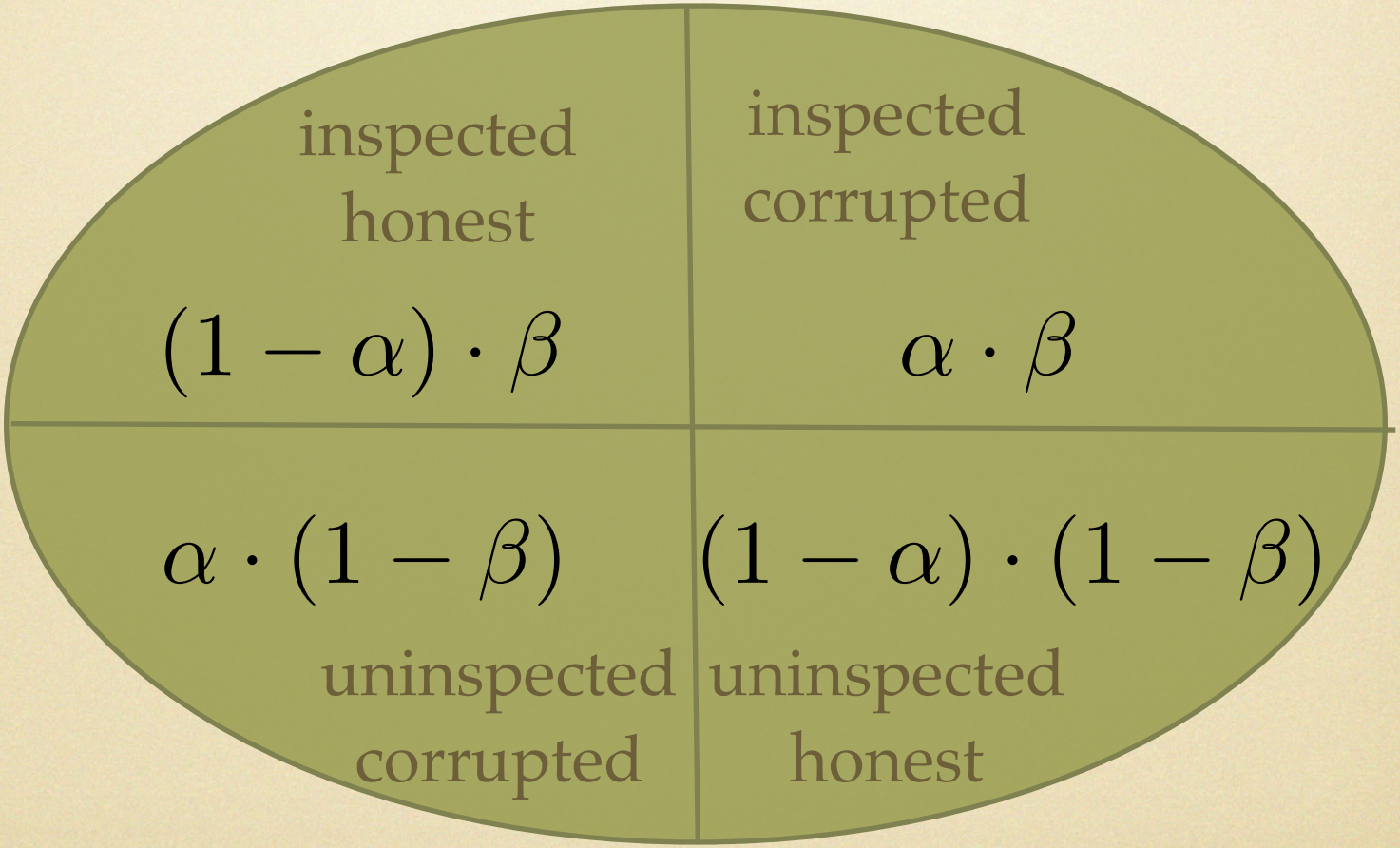
$$\gamma = \frac{1-\alpha}{2}$$

achievable via standard crypto:

$$\gamma = \frac{1}{2} - \alpha$$

a gap of $\frac{\alpha}{2}$

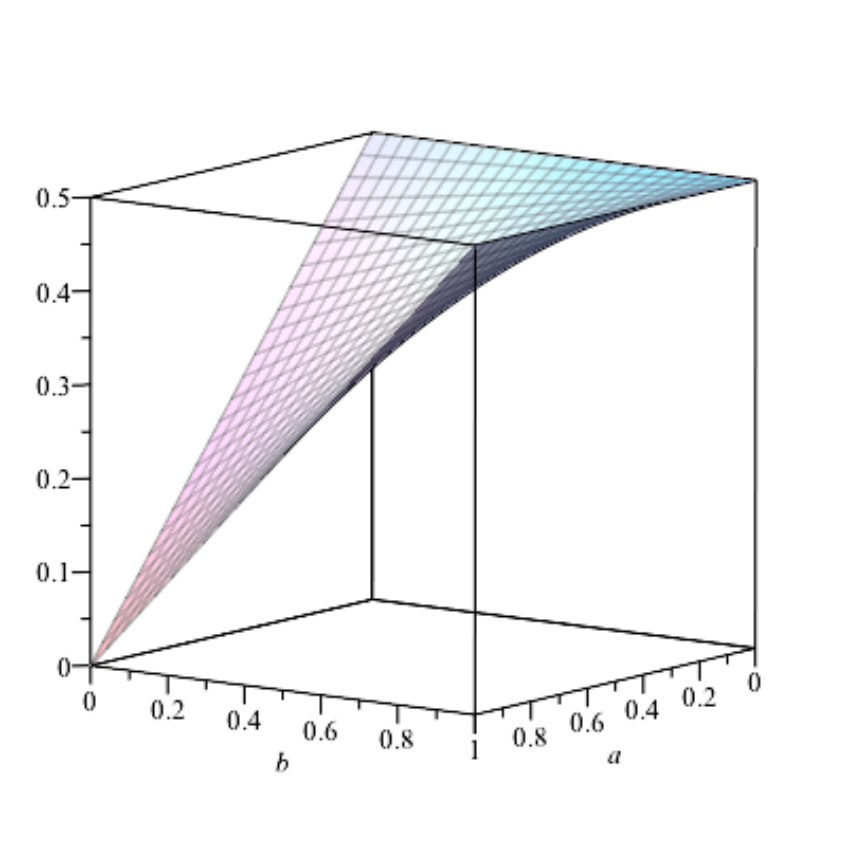
In general

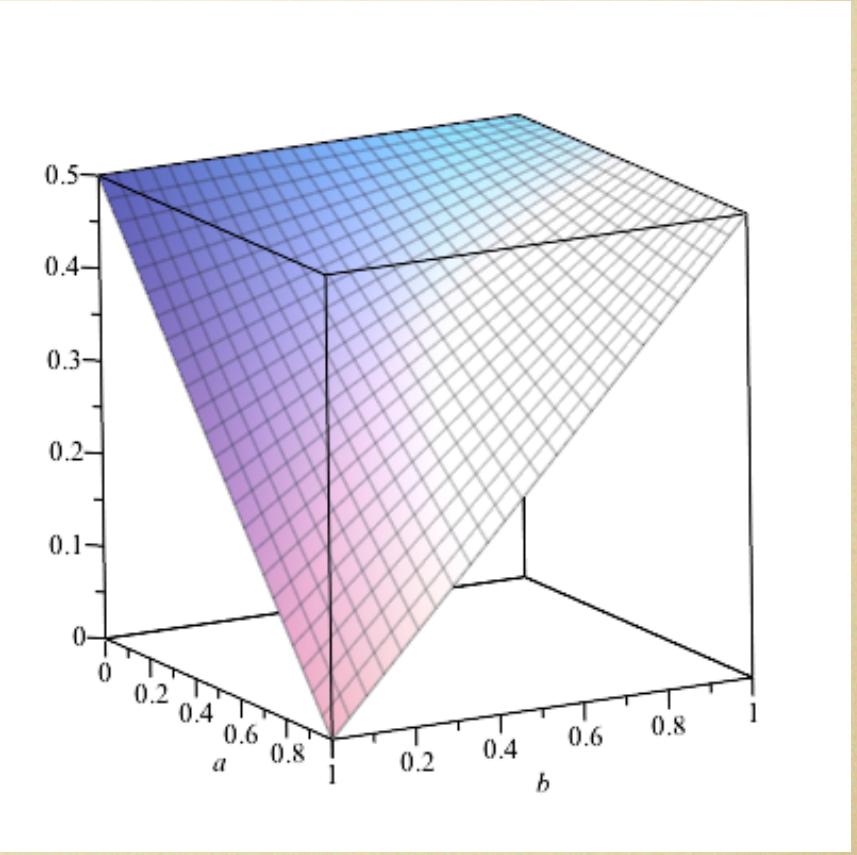

inspected
honest
$(1-\alpha)\cdot\beta$
inspected
corrupted
$\alpha\cdot\beta$
$\alpha\cdot(1-\beta)$
uninspected
corrupted
$(1-\alpha)\cdot(1-\beta)$
uninspected
honest

The ‘dream’ bound

on corruption resiliency of the total system

$$\gamma < (1 - \alpha + \alpha \cdot \beta)/2$$





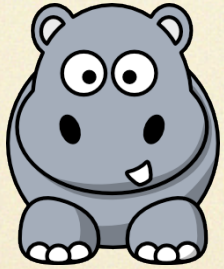
A Fundamental Question

- How to harness the power of any remaining honest servers that are lost in a pool of corrupted ones?
  - are the dream bounds of corruption resiliency approximable without requiring the SP to invest a lot ? (i.e., using a high β)

our new crypto protocols

- we show that, under reasonable system assumptions, there is a way to utilize the honest servers even though we don't know where they are!

The 2-tiered model

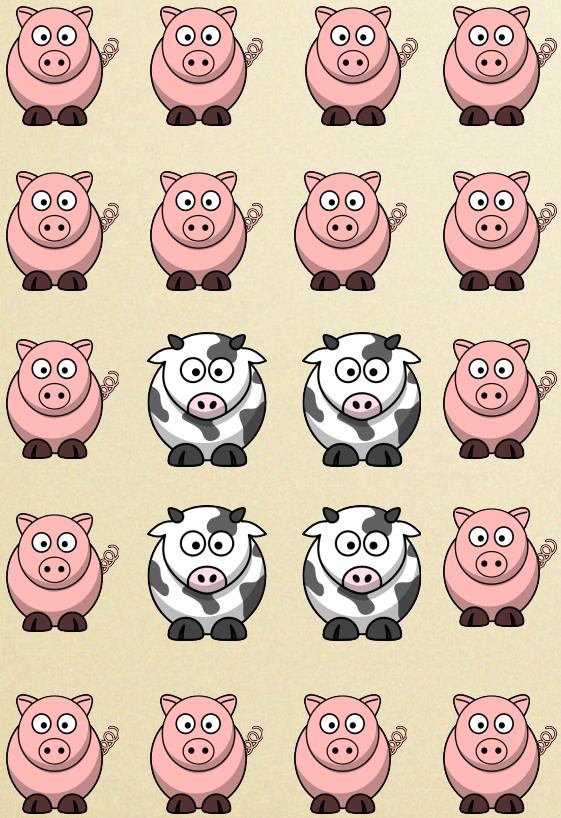


Consider an SP that has two kinds of servers:

cows : always good

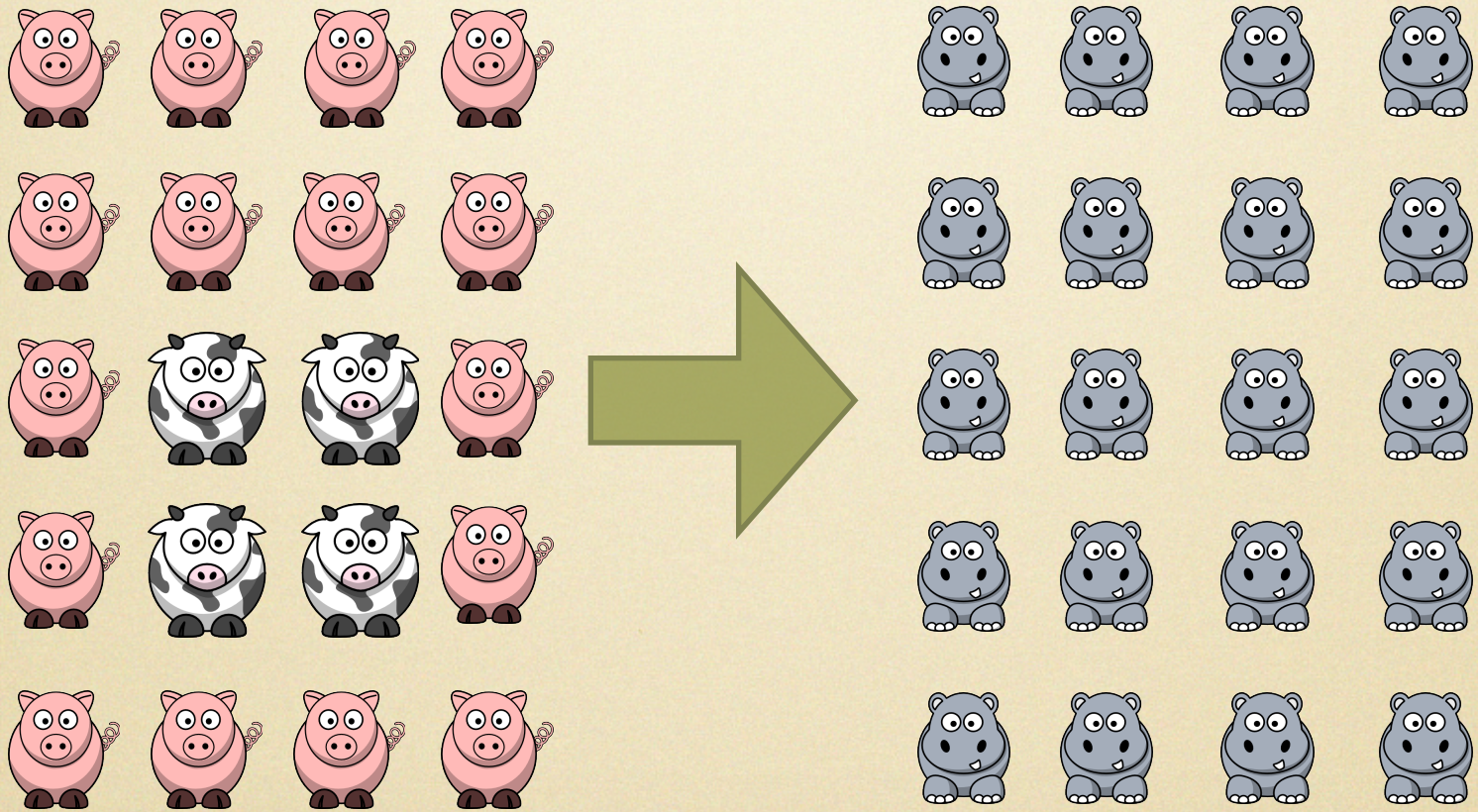
pigs : sometimes good

The two-tiered model for MPC



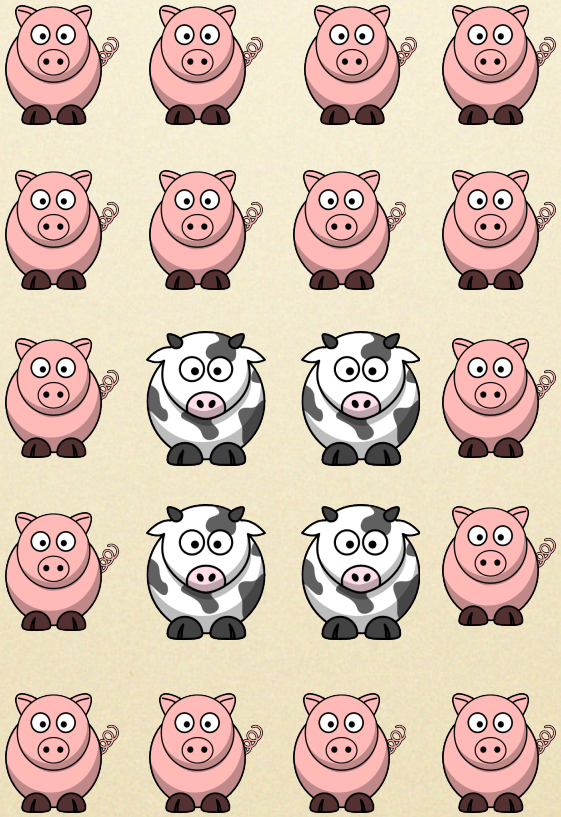
Assumption: servers are indistinguishable in the eyes of the adversary

The two-tiered model for MPC



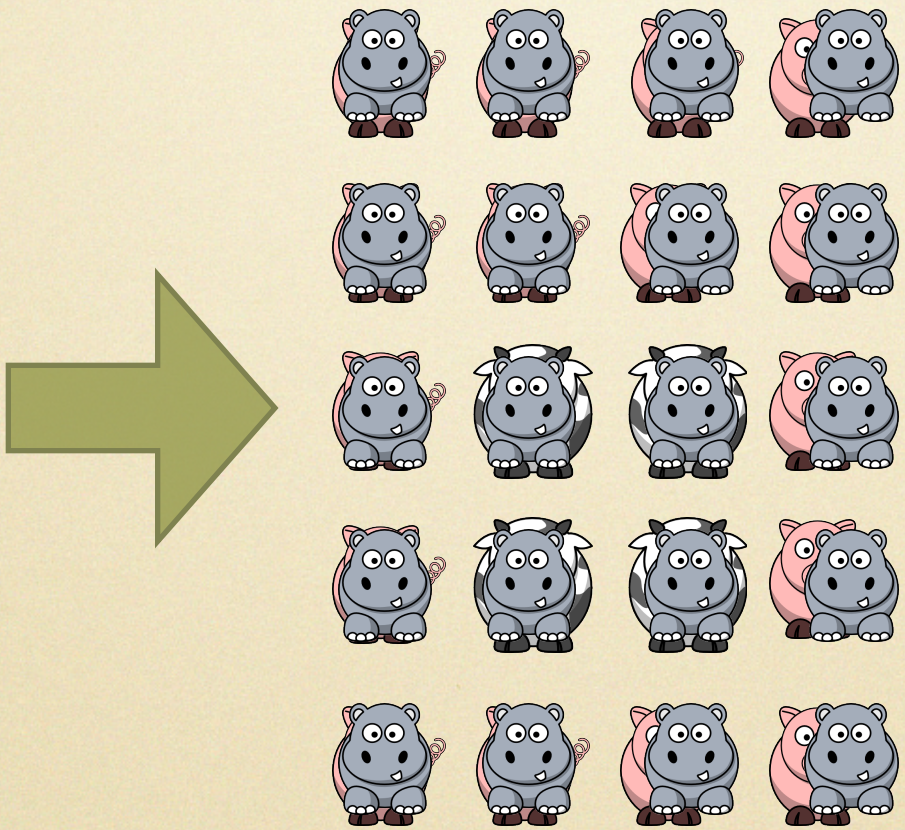
Assumption: servers are indistinguishable in the eyes of the adversary

...then



Corruption is like picking balls from an urn without replacement.

...then



Corruption is like picking balls from an urn without replacement.

Main Technical Lemma & Corollary

Starting with n servers (pigs + cows)
it is possible via a protocol that uses anonymity to approximate the maximum
"corruption resiliency" of the system, by utilizing only $\omega(\log n)$ cows.

Corollary. The dream bounds of corruption resiliency are attainable asymptotically assuming server anonymization.

Thank you

Securing services running over untrusted clouds : the two-tiered trust model

Aggelos Kiayias (U. Athens & U. Conn)

Joint work, **Juan Garay, Ran Gelles, David Johnson, Moti Yung**
(AT&T – UCLA – AT&T – Google)