Preventing Cryptographic Key Leakage in Cloud Virtual Machines

Erman Pattuk
Murat Kantarcioglu
Zhiqiang Lin
Huseyin Ulusoy

Move to Cloud Computing

Move to Cloud Computing

Security Concerns

- To authenticate themselves to regular public
  - SSL/TLS protocols
  - Cryptographic keys
- Cloud service providers use Virtual Machine Monitors
  - To realize logical isolation
- A company's VM(s) are located on the same physical machines as other companies
  - Potential threat!!!

Side-channel Attacks

- Ristenpart et al. (2009) attacked a target cloud VM
  - Co-located his VMs with the defender's
  - Executed cross-VM side-channel attack
  - Captured crude data (e.g., aggregate cache usage)
- Zhang et al. (2010) used the same co-location technique
  - Extracted El-Gamal keys
- A wide-variety of side-channel attacks
  - May be applicable to cloud

Many ideas to protect against side-channel attacks

- HomeAlone (Zhang et al., 2011)
  - Co-residency check to understand where a victim VM is located
  - Aims physical isolation
  - May not be applicable to modern cloud infra
- HyperSafe (Wang et al., 2010)
  - Aims to preserve hypervisor integrity
  - We assume that is provided in cloud (e.g., Amazon)
- Randomization based prevention techniques
  - Karlof et al. (2003), hidden markov model based attack to break down randomization

Potential other attacks ??

- Side channels may not be the only type of attacks.
- Software bugs may be used to disclose keys as well.
  - Hearthbleed bug?

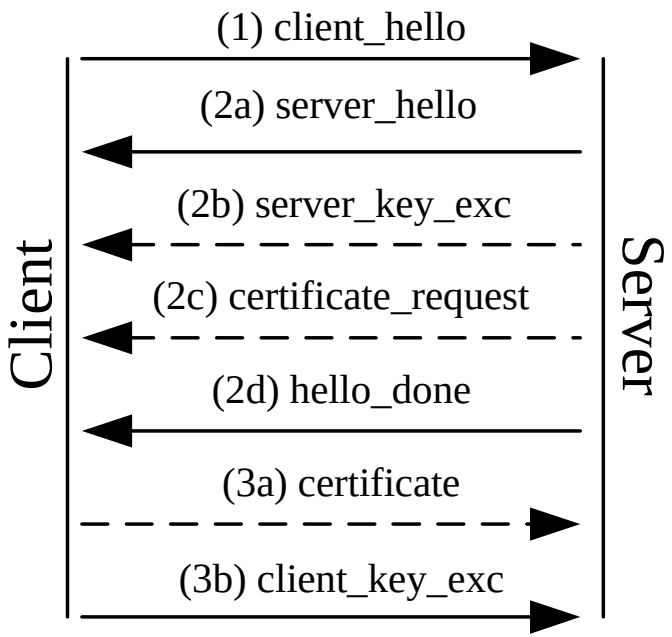
Hermes – Key Idea

- Partition private keys into n shares
  - Threshold cryptography
  - Shamir secret sharing
- Create multiple VMs
  - Give one share to each VM
  - Collaborate to decrypt/sign with the private key
- Re-share the private key periodically
  - Some shares may be captured over time
  - Create new shares that are independent

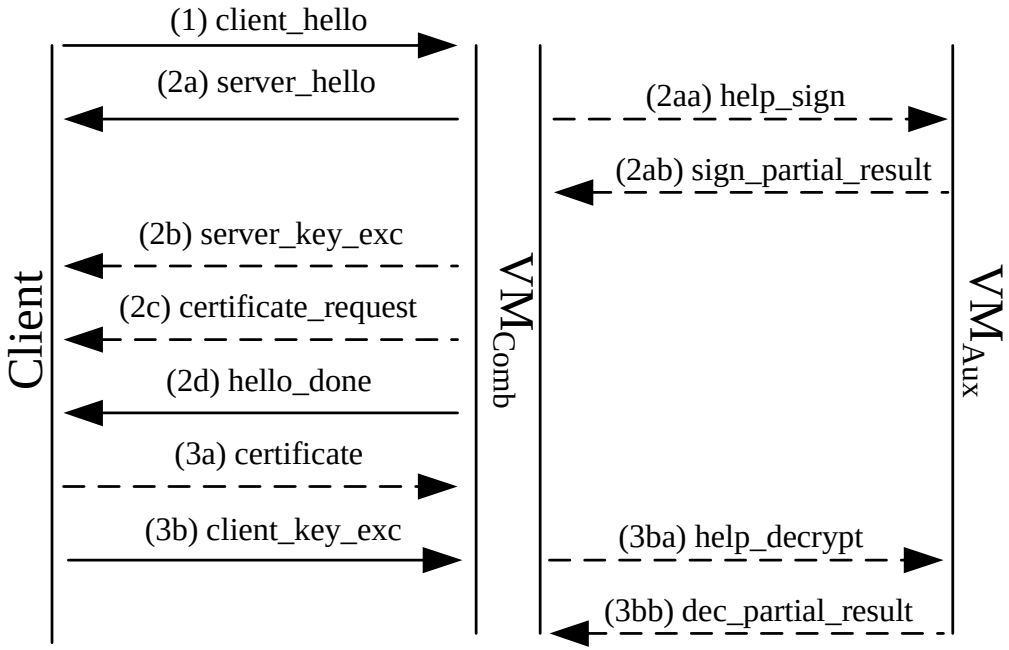
Background

- Hermes runs in two modes
  - Distributed-RSA (D-RSA): Additive secret sharing
  - Threshold-RSA (T-RSA): Shamir secret sharing
- Given an RSA key pair e,d for modulus n:
- Additive secret sharing:
  - $d = d\_1 + d\_2 + \ldots + d\_L \bmod \Phi(n)$
  - Need all shares to exponentiate a message
- Shamir secret sharing (Shoup, 2000):
  - d is embedded into K-1 degree polynomial P
  - Each party id ε {1, ..., L} is given P(id)
  - Need K shares to exponentiate a message

Hermes – SSL Enhancement

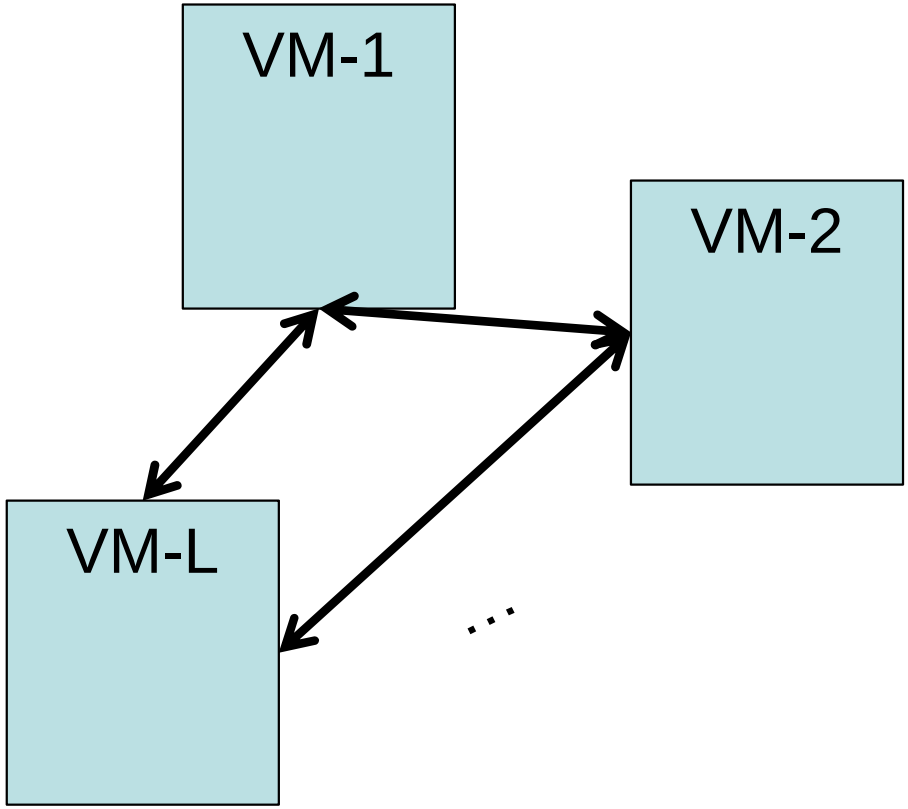


Regular SSL

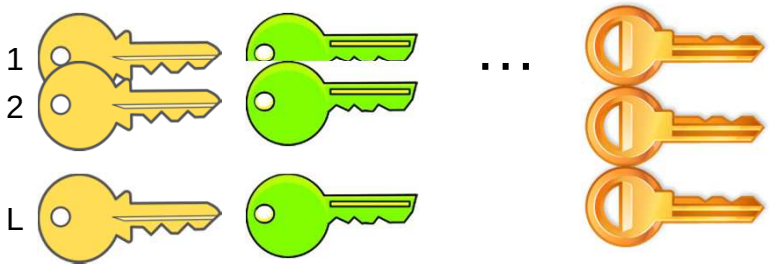


Enhanced SSL

Hermes - Initialization

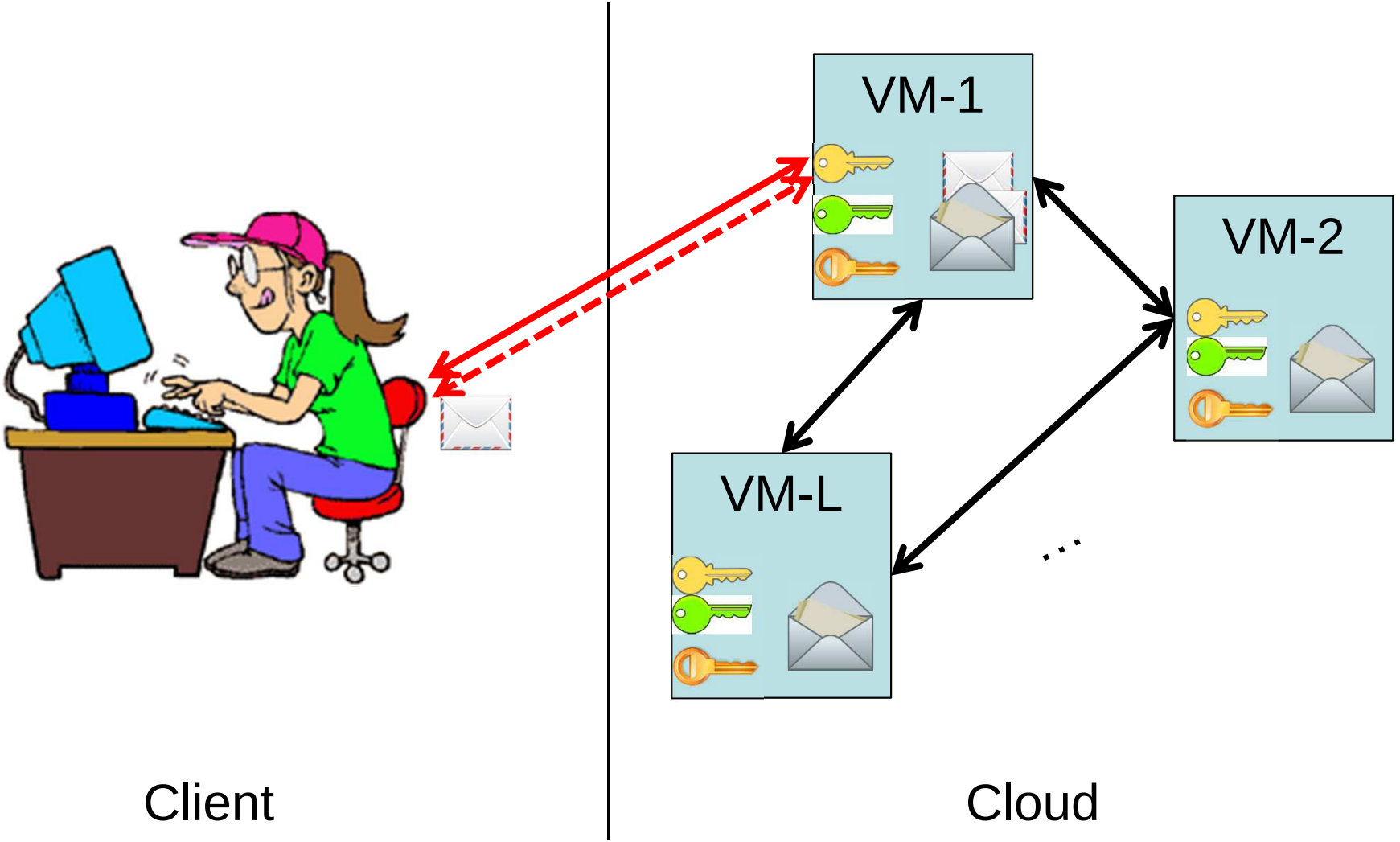


Cloud

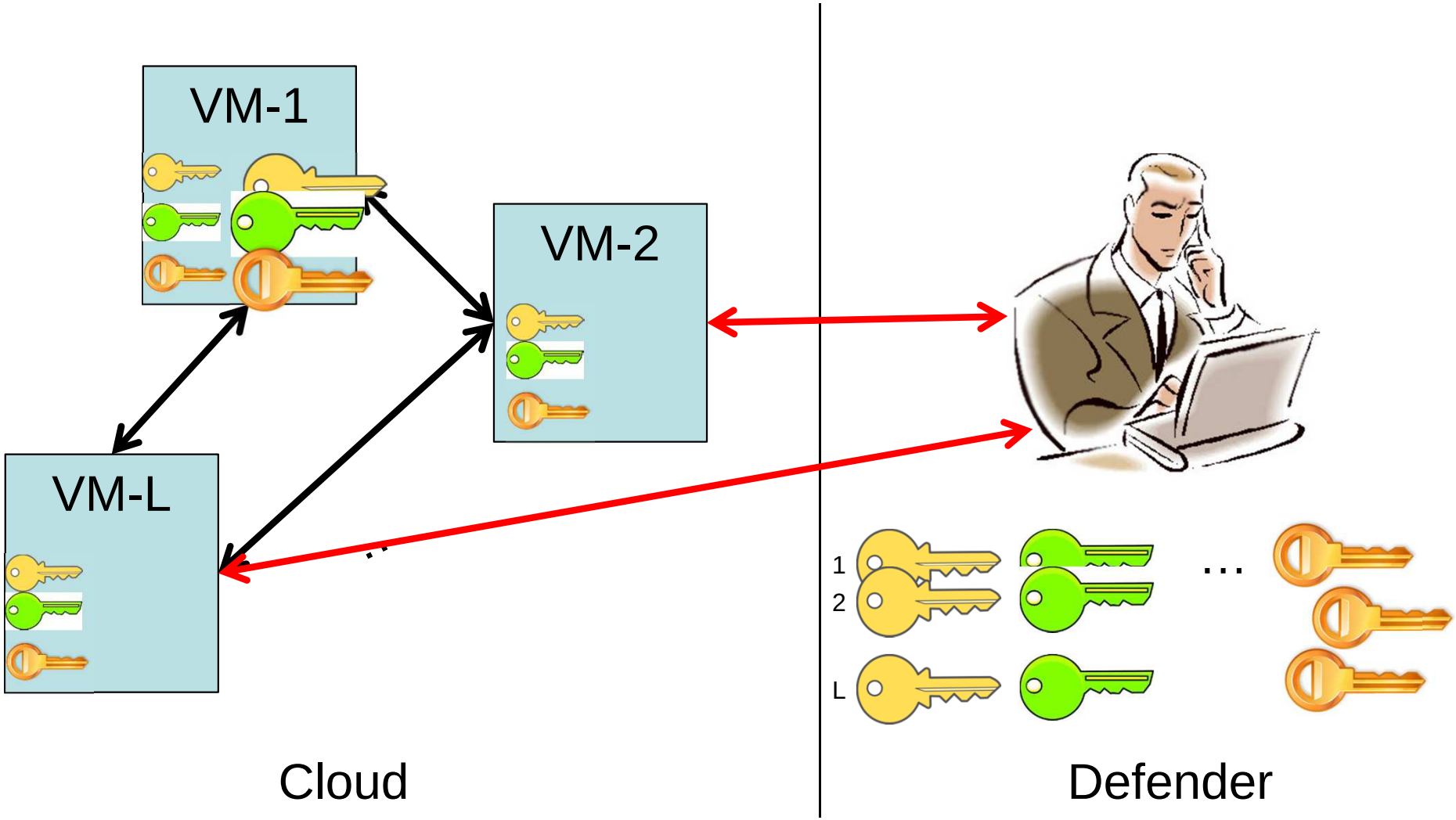


Defender

Hermes – Client Connection



Hermes – Key Re-sharing



Experiment Setup

- Used Hermes in three applications:
  - Micro-benchmarking: Clients connect and die
  - Web-server: Clients retrieve a web page
  - Mail-server: Clients retrieve a mail box
- Initiated 10 VMs in Amazon EC2
  - 1 m1.xlarge with 4 virtual CPU as combiner VM
  - 9 m1.small with 1 virtual CPU as auxiliary VMs
- Clients:
  - Apache HTTP benchmarking tool and JMeter to connect
  - From an IBM x3500m3 server in our campus
- Implemented as a shared library in OpenSSL

Micro-Benchmarking Fixed Parties L=10

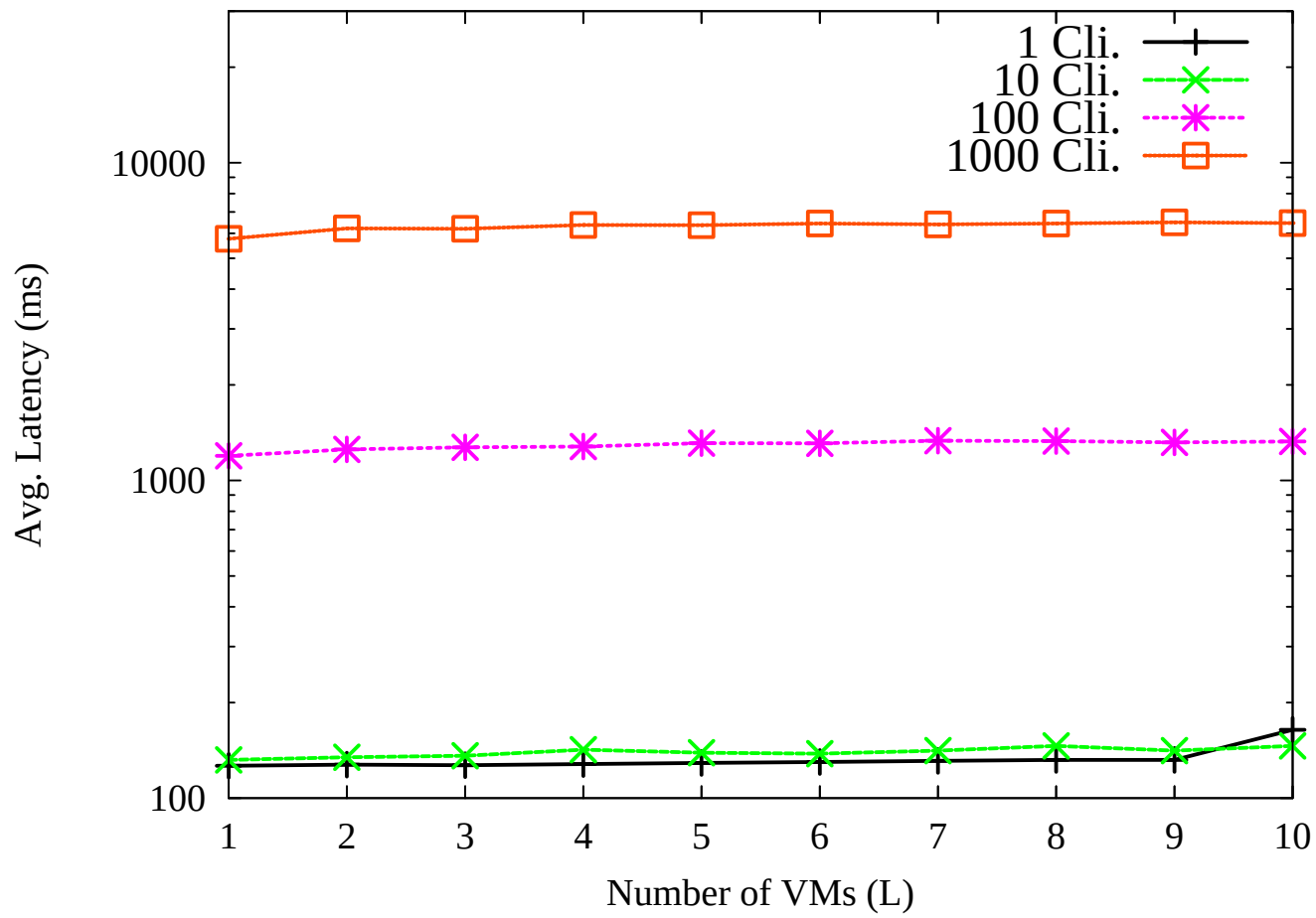
		Setup								
		(10,2)	(10,3)	(10,4)	(10,5)	(10,6)	(10,7)	(10,8)	(10,9)	(10,10)
1 Clients	Total	12.06	12.27	13.44	11.57	16.10	17.94	16.80	19.80	13.76
	Network	4.97	5.14	4.89	2.75	5.07	5.59	5.29	1.15	1.56
	Combine	0.52	0.56	0.58	1.82	2.36	1.56	2.44	2.20	1.93
10 Clients	Total	19.78	23.22	28.14	45.87	39.48	48.99	49.97	60.70	52.82
	Network	9.72	10.15	10.19	14.29	16.30	23.15	27.17	34.03	14.09
	Combine	1.27	1.07	1.26	2.22	2.42	2.64	3.09	2.81	2.23
100 Clients	Total	54.90	71.07	88.31	187.67	130.17	163.24	182.12	206.00	269.73
	Network	11.77	25.27	37.77	122.96	69.74	84.05	82.96	121.32	113.15
	Combine	1.24	1.62	1.74	2.01	2.15	2.34	2.80	3.01	2.28
1000 Clients	Total	318.24	418.07	435.98	928.75	653.12	642.48	877.42	995.89	1174.54
	Network	88.12	123.72	130.21	202.08	196.29	212.05	214.20	216.97	233.41
	Combine	1.50	2.20	1.85	1.96	2.08	2.52	2.82	3.24	2.62

- Overheads could be significant due to network connection
  - (2-6 times) slower
- Can be impacted by Amazon workloads

Micro-Benchmarking Fixed Shares k=2

		Setup								
		(2,2)	(3,2)	(4,2)	(5,2)	(6,2)	(7,2)	(8,2)	(9,2)	(10,2)
1 Clients	Total	8.96	12.03	11.23	11.97	12.37	13.47	12.16	9.58	12.06
	Network	2.77	4.77	4.82	5.28	5.10	4.77	5.42	4.76	4.97
	Combine	1.78	2.13	0.90	1.30	1.35	0.51	0.53	0.55	0.52
10 Clients	Total	31.74	33.45	23.77	25.34	23.57	20.18	18.81	19.26	19.78
	Network	19.42	19.10	11.00	9.78	9.62	8.20	10.49	9.26	9.72
	Combine	1.95	2.26	1.52	1.47	1.26	1.50	1.29	1.33	1.27
100 Clients	Total	179.01	164.65	95.30	82.50	80.98	66.95	73.26	58.08	54.90
	Network	121.05	95.84	52.52	38.03	30.00	25.90	25.93	25.26	11.77
	Combine	2.16	2.19	1.82	1.92	1.42	1.59	1.72	1.15	1.24
1000 Clients	Total	640.40	665.95	548.22	504.12	450.09	340.10	350.46	320.84	318.24
	Network	210.36	197.43	150.84	123.75	60.88	55.09	59.19	46.50	88.12
	Combine	2.26	1.93	1.91	1.91	1.55	1.41	1.42	2.36	1.50

Mail Server Results



Optimization: Key Idea

- Although keys are distributed, adversary may capture some keys
- Choosing parameters is critical:
  - L: Number of VMs
  - K: Number of shares needed
  - τ: Re-sharing frequency
- Choose optimal values based on 3 aspects:
  - Security: How likely the adv. will succeed?
  - Cost: How much is to purchase L VMs
  - Performance: What should be K

Optimization: Security Aspect

- Probability of a successful attack on a VM
  - Exponentially distributed random variable
  - Θ: Mean time to succeed

$$f(t) = \begin{cases} \frac{1}{\theta} e^{-t/\theta} & \text{if } t > 0 \\ 0 & \text{otherwise} \end{cases}$$

- Probability to succeed in a re-sharing period

$$F(\tau, \theta) = \int_0^{\tau} f(t).dt = 1 - e^{-\tau/\theta}$$

- Capturing k shares in a period

$$Sec(l, k, \tau, \theta) = \sum_{i=k}^{l} \binom{l}{i} (1 - e^{-\tau/\theta})^i (e^{-\tau/\theta})^{l-i}$$

Optimization: Other Aspects and Objective

- Cost increases linearly with number of VMs:
  - β: Unit cost of a VM

$$Cost(l) = l.\beta$$

- Performance aspect depends on:
  - The particular application (e.g., Web or mail server)
  - Performance metric (e.g., latency)
- Objective:

$$\begin{aligned} &\text{minimize:} && Sec(l,k,\tau,\theta) \\ &\text{subject to:} && Cost(l) \leq L_{cost}, Perf(l,k) \leq L_{perf} \\ & && l \geq k > 1, \tau > 0 \end{aligned}$$

Optimization: Application to Micro-Benchmarking

- Applied optimization to Micro-Benchmarking
  - 100 clients
  - Performance metric is latency
  - 5 second re-sharing period
- Used experiment results up to L=10 VMs
- Trained linear regression on latency results

$$Perf(l,k) = c_0 + c_1.l + c_2.k + c_3.(l/k)$$

  - c_1 = -18, c_2 = 31, c_3 = 7
- Calculated optimal values for
  - Latency between 50-200 ms
  - Annual budget between $1820 - $14560 for L=[2,16]

Optimization: Application to Micro-Benchmarking

	$\theta = 600$		$\theta = 3600$	
L_{cost}/yr	Conf.	Sec()	Conf.	Sec()
\$1820	$(2,2)$	$6.8 \cdot 10^{-5}$	$(2,2)$	$1.9 \cdot 10^{-6}$
\$3640	$(4,3)$	$2.2 \cdot 10^{-6}$	$(4,3)$	$3.7 \cdot 10^{-8}$
\$7280	$(8,5)$	$2.1 \cdot 10^{-9}$	$(8,5)$	$2.8 \cdot 10^{-13}$
\$14560	$(16,10)$	$1.1 \cdot 10^{-17}$	$(16,10)$	$2.1 \cdot 10^{-25}$

	$\theta = 600$		$\theta = 3600$	
L_{perf}	Conf.	Sec()	Conf.	Sec()
50 *msec*	$(16,6)$	$2.4 \cdot 10^{-9}$	$(16,6)$	$5.6 \cdot 10^{-14}$
100 *msec*	$(16,8)$	$2.7 \cdot 10^{-13}$	$(16,8)$	$1.7 \cdot 10^{-19}$
150 *msec*	$(16,10)$	$1.1 \cdot 10^{-17}$	$(16,10)$	$2.1 \cdot 10^{-25}$
200 *msec*	$(16,11)$	$5.4 \cdot 10^{-20}$	$(15,11)$	$4.9 \cdot 10^{-29}$

Summary

- Secret sharing can be used to protect public keys against cross VM attacks and more..
- Performance overhead can be very low for CPU bound applications
- Optimal parameters can be found using multi-objective optimization framework

Questions