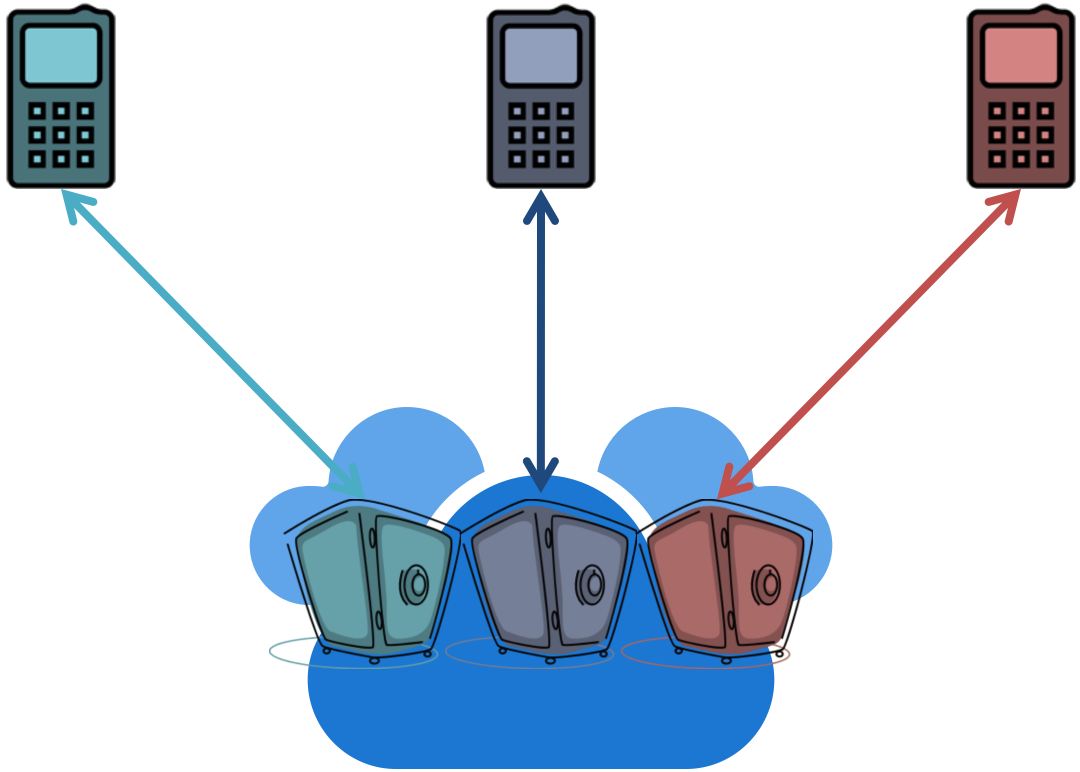
Architectural Principles for Secure Multi-Tenancy

John Linn, Office of the CTO, RSA, The Security Division of EMC

John Field, Office of the CTO, EMC

Also adapting prior content by Burt Kaliski

DIMACS Workshop in Systems and Networking Advances in Cloud Computing, December 2011



multi-tenancy

hosting by a provider of **more than one tenant at the same time**

Essential to effective cloud computing, by enabling resources to be shared in a secure and cost-effective fashion

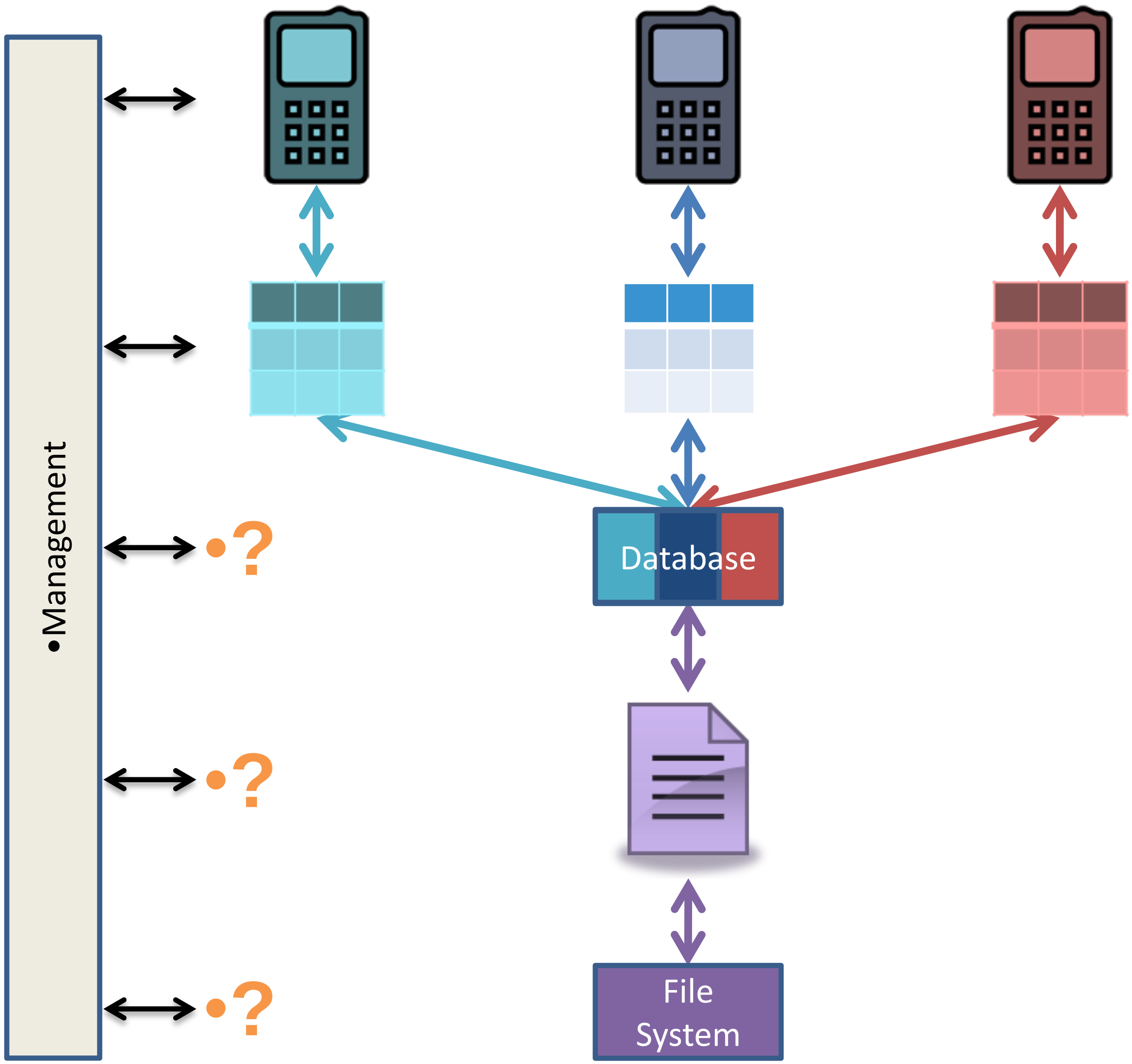
Objectives and Principles

- Layered approach, applicable to alternate cloud models
  - Individual components comprising the infrastructure on which clouds are built
  - Provider cloud-level infrastructures, layered on interconnected sets of components
  - Tenant clouds, layered on infrastructure elements operated by the provider and tenant
- Providers manage the first two layers, allowing tenants to implement the third
  - This talk examines both provider and tenant perspectives
- Subscribers must place trust in providers per Service Level Agreements (SLAs), but the trust level shouldn't be (or need to be) unlimited

Constructing Multi-Tenant Infrastructures

- A secure multi-tenant (MT) infrastructure can combine two types of components
  - MT-capable: components that provide explicit multi-tenancy support and are trusted to process and segregate information belonging to multiple tenants
  - Non-MT: components unable to enforce such segregation, that must not receive multiple tenants' information in a form that could enable leakage across tenant boundaries
- Can segregate non-MT components and across tenant boundaries via physical, network-based, and/or crypto methods
- Generally, we expect MT-capable components to evolve upward from platforms, hypervisors, and selectively into the application stack
  - Trustworthy components require trustworthy supporting layers
  - Trusted MT-capable hypervisors are fundamental components for cloud architectures

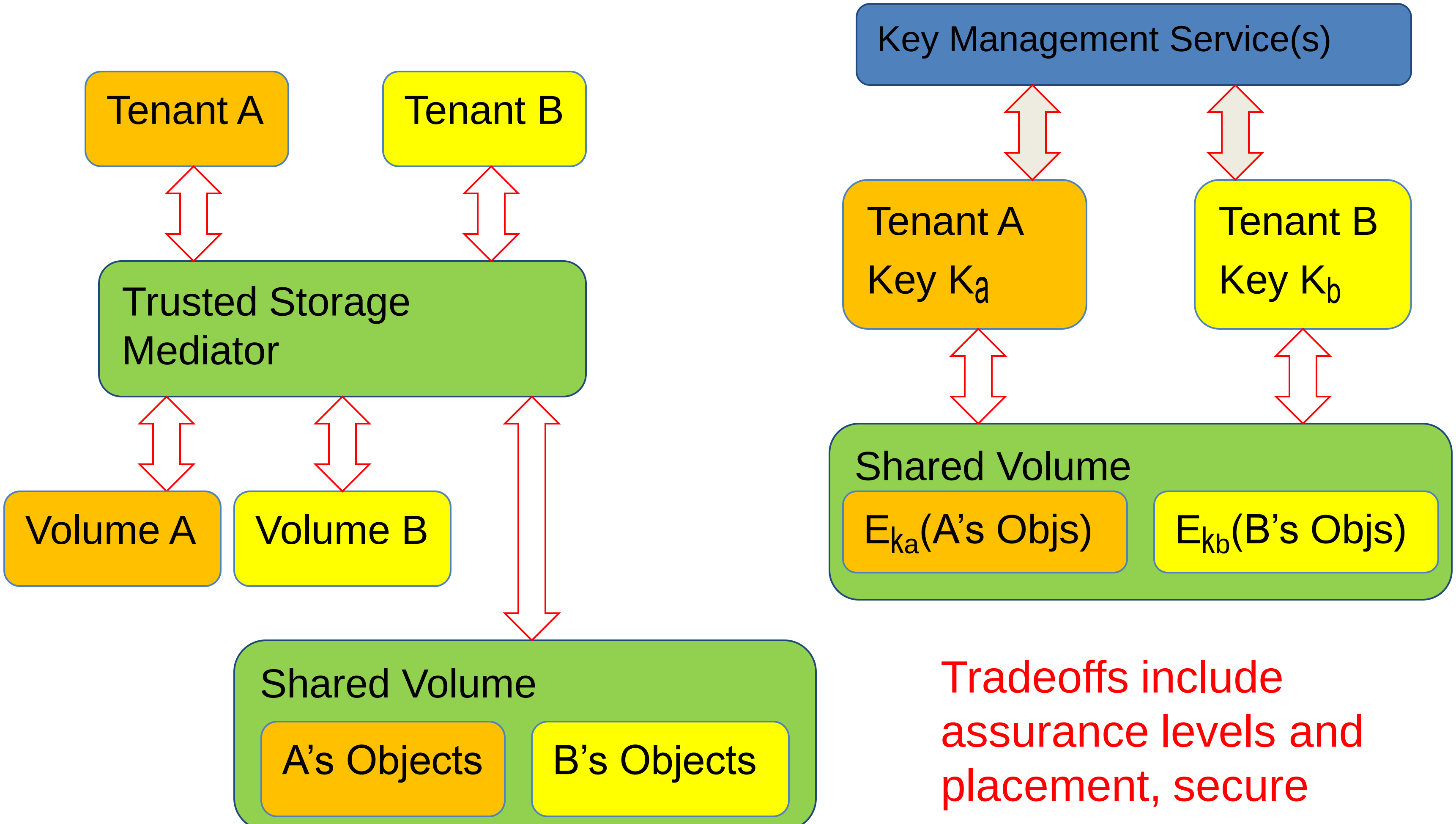
Multi-Tenancy: Consolidated Database Example



Securing Storage Elements

- Storage resources must be restricted to appropriate tenants
  - Explicitly by MT-capable mediators
  - Implicitly by configuration, so that non-MT components can access only devices and volumes corresponding to their tenants
- Need controls on access to containers and their information objects
  - Container-level access typically mediated by cloud provider
  - Object-level access mediated by provider and/or by tenant
- Resources must be clean when provided, persistent and protected while in use, securely cleaned when no longer needed
  - Challenges: ensuring confidentiality, ongoing integrity, guaranteed erasure

Example MT Storage Alternatives



Tradeoffs include assurance levels and placement, secure erasure

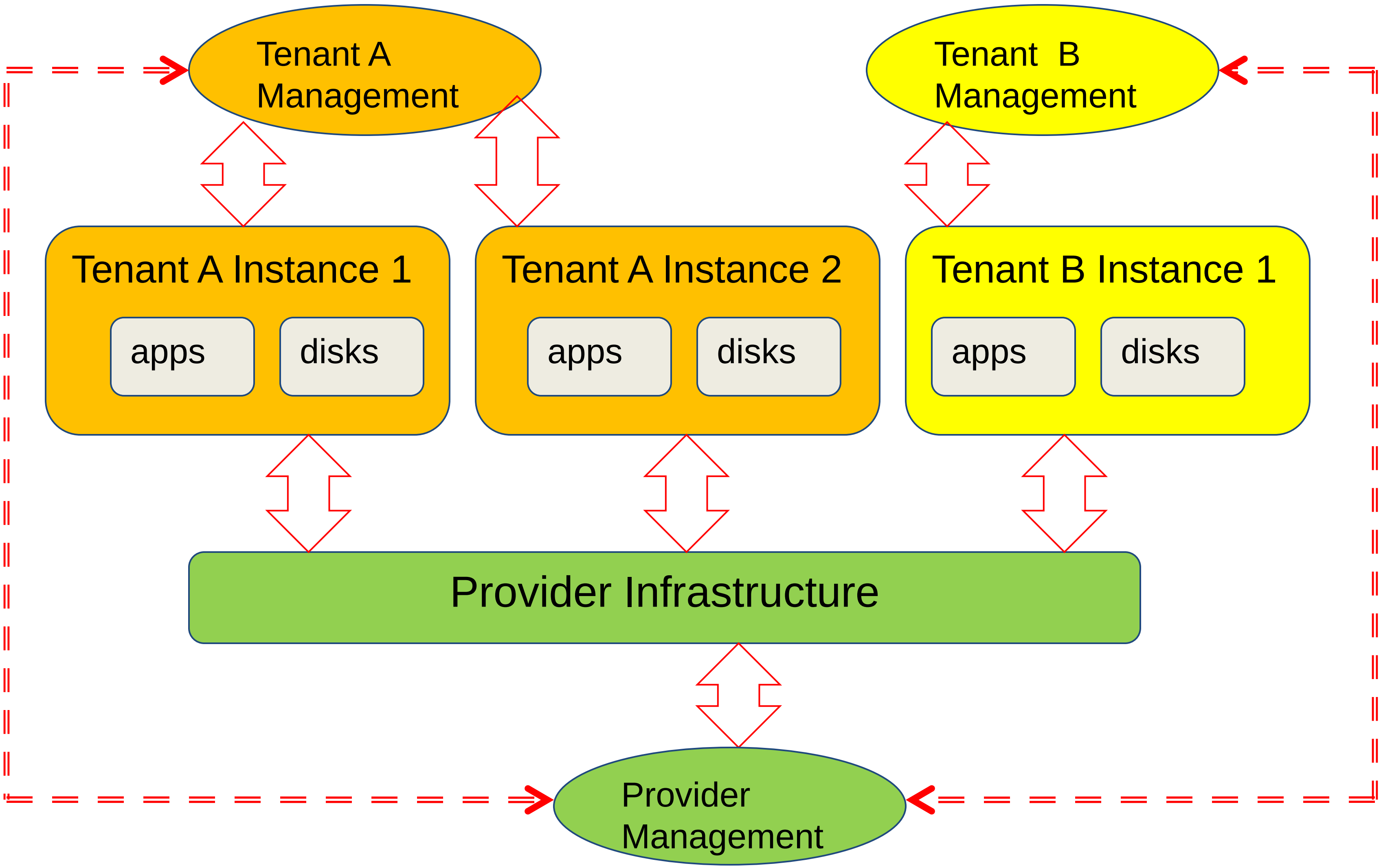
Securing Provider Clouds: Isolation and Integrity

- Normally, a tenant’s instance should be enclosed in a virtual perimeter, with elements and communications paths disjoint from other tenants and instances
  - Underlying physical resources can usually be shared
  - Selective external connectivity (default deny) is usually needed; providers can offer monitoring and filtering services
- VMs should be identified uniquely and associated with their instances; cryptographic credentials are attractive
- Security SLAs can specify assurance levels and/or methods to be applied
  - Verifiability by tenant and/or third party is important
  - Providers can offer scanning services to subscribers
- Challenge: effective integrity verification of dynamic objects

Securing Provider Clouds: Audit and Compliance Support

- Effective cloud monitoring requires data from multiple components and layers; both providers and subscribers have monitoring needs
  - Cloud-level attacks are relevant to subscribers, and tenant-level activity is relevant to a cloud's overall security posture
- Flow of monitoring information should be constrained by authorization and privacy policies
- Cloud infrastructure should allow for discovery, measurement, and recording of proper, secure resource configuration
- Conflict: providers prefer opacity, subscribers seek transparency
  - Possible rendezvous point: neutral independent parties

Tenant-Provider Relationships



Provider-side Conclusions

- Layered architecture is fundamental
  - Monitoring and management must span layers coherently, while satisfying authorization and privacy policy
- Should allocate MT segregation functions to appropriate trusted components, building up from platforms and hypervisors
  - May also need higher-level MT, particularly to achieve application-level or user-level protection granularity
- Successful cloud providers will be attractive attack targets, and need to protect themselves and their subscribers
  - Providers must protect themselves and tenants against collateral damage
  - Cloud provider security challenges and responsibilities may exceed those of conventional data center operators
- Overall challenge: enable providers to construct efficient operational environments, where their tenants' trust in those providers can be constrained

The Tenant View

- Architectural considerations for the tenant:
  - Service Level Compliance
  - Self-service Considerations
  - Integrated Monitoring
  - Limiting Trust

Tenant View: Service Level Compliance

- Providers and tenants are assumed to be mutually untrusting.
- Contract between the provider and the tenant is embodied in a Service Level Agreement (SLA).
- Verifying compliance with SLA requires the provider to surface specific evidences.
  - However, provider will not disclose infrastructure configuration details.
- Verification requires agreement on how to validate the “what” represented by the SLA.
- Challenge: What evidence constitutes sufficient proof of the SLA?

Tenant View: Self-Service Administration

- Tenants require autonomous control of the tenancy, with minimal dependency on the provider.
  - Not just a matter of convenience; also ensures tenant privacy.
- Desire consistent management model across all resource types
  - Compute, Network, Storage.
- Tenants must also anticipate the need to perform self-service across multiple Tenancies.
  - Consistency improves efficiency, and reduces risk.
- Challenge: Federated and delegated authentication, satisfying provider and tenant assurance requirements.

Tenant View: Self-Service Resources

- All tenant-visible units of resource management must be logical, not physical.
- Enable those objects to be scoped to (nested in) a container abstraction.
- Enable recursive delegated administration capabilities at the container layer.
- Implement out-of-band monitoring of management activities
  - Verify actual state of the system remains in compliance across management state changes, across tenancies.

Tenant View: Integrated Monitoring

- Monitoring information from providers must be integrated with tenant systems.
  - Lack of integration increases costs, and introduces delays in detection, response, and remediation.
- Providers should offer tenants appropriate visibility into provider infrastructure operations.
  - Service State Information Model
  - Automation in incident response
- Challenges: service modeling, and protocols for automation of incident response.

Limiting Trust In Providers

- Tenants may deter and/or mitigate the risks in trusting the provider through compensating controls
  - Strong authentication of all actors, appropriate separation of duties, comprehensive auditing, and integration with Tenant's security monitoring.
- Provider environments should enable tenant-controlled confidentiality and integrity.
  - Tenants may choose not to share encryption keys with provider.
- Trust in availability is necessary: provider is a potential single point of failure for a tenancy.
- Challenges: Ensuring availability, requisite trust properties, and providing corresponding controls.

Research Challenges and Conclusions

- Need enhanced techniques and assurance for trustworthy computing
  - When other tenants aren't trusted
  - When a platform isn't fully trusted
  - When the platform's operator isn't fully trusted
- Need effective means to validate integrity of items that are intended to change
- Want effective means to process data without exposing its information
- Modeling and verifying service relationships.