Nikolai Joukov, Ph.D.

Enterprise IT Discovery
Content

- Introduction
  - this chart, sign-up

- Enterprise IT
  - real world

- IT discovery
  - manual, per-host, based on network traffic, services-friendly, physical
  - CMDBs
    - how to keep the information

- reading list
Example Data Center Buildings

Microsoft

IBM

CyberBunker
Inside of a datacenter

racks

cold-hot aisles

storage

Raised Floor Airflow

Servers are usually Coolest at the Bottom of the Rack
A Part of Google’s “Cloud” (Warehouse Scale Computing)
Mainframes and other scalable systems

- Not all workloads can be executed on 2-4 CPU servers
- E.g., Large enterprise databases may need 64 or more CPUs in one server
- These servers support various hardware-based virtualization options
Enterprise Applications

- You will hear “Application” everywhere and during all lectures of this course.
- Application is a high-level view on IT based on the performed business function.
- Examples: travel expense reimbursement application, student grading application.
- Google search is a large distributed application too.
- One application can consist of one or hundreds of servers.
- Some servers or even objects like databases are shared by multiple applications.
- Applications have owners.
- Applications have dramatically different criticality for enterprises.
Enterprise Storage Components

- Almost any component can interact directly with another component
- e.g., a database can store data on a file system or directly on a disk partition

Block level unit of data is sector(s)
Linux Server DB2-to-Storage Dependencies Example

- DB2, two instances, databases
- LVM install, volume groups, volumes
- SCSI disk, partitions
- Ext3 mounts
- NFS mounts
- Unused, not partitioned IDE disk
- Another SCSI disk and partition
- DB2 on another server that we did not scan
- NFSD on another server that we did not scan
- DB2 on another server that we did not scan
- NFSD on another server that we did not scan
Storage Area Networks

- Block-level networks to transfer data between storage devices to servers
- Dedicated networks
- Almost always each server has two redundant links to the storage controllers
- Each server adapter and each storage controller port has a World Wide Port Name (WWPN)
  - Relinking servers and storage requires updating port mappings
- SANs are zoned for security
- Logical Unit Number (LUN) is the unit of data accessed by the servers
Enterprise Storage Systems

- Fully redundant
- Each hard disk has two data ports
- Each such port is connected to a separate server inside of the storage controller
- Disks are RAIDed and have hot spares
VMware ESX Client VM (left) and Server (center)
Application-level view (a real enterprise environment)
server-to-server dependencies (infrastructure)
Middleware Clusters

- All important enterprise middleware (web servers, databases, etc) are clustered or are supposed to be clustered
- Some clusters are middleware clusters
  - e.g., HTTP servers and Java EE servers can have clusters per-url or per Java application
- Most clusters are fail-over
OS/hardware clusters

- Some clusters are OS/hardware
  - e.g., AIX, Linux, Windows OS and Veritas Clusters
- They almost always assume shared storage clusters via Storage Area Networks (SANs)
- They can be integrated with middleware and monitor if say a database is servicing requests
- Typically have multiple heart-beat loops
- Clusters have resources:
  - IP addresses
  - DNS names
  - Software instances
  - ...
Clusters across datacenters

- Data gets replicated across datacenters
- Reads are no problem but writes have to be synchronized over long distances
- Typically replication is done via storage replication or database replication
- Non-enterprise databases have problems with this and thus enterprises pay for expensive Oracle, DB2, Sybase, etc. licenses
Networking infrastructure example

- network zones
- multiple networks per server
- dedicated links
Enterprise Business Models

Value Network Example
People and Processes

- You need permission to do anything
- Specific people are responsible for databases, policies, servers
  - need to get permission and buy-in from all of them to change anything
- Processes are poorly documented so need to talk to people
Enterprise IT: Abstractions

- **data center** (enterprises can have dozens of them)
  - building management
  - cooling
  - electrical equipment
  - inter-data center network links
  - racks
  - servers
  - networking
  - which virtual machine is located on which physical server

- **application** (aka business application)
  - web server
  - java program
  - database

- **business** (non IT people think in these terms)
  - business units
  - lines of business
  - supply chains
  - value networks

- **people**
  - architects, sysadmins, DBAs, application owners, security teams, etc.

- **processes and policies**
  - how to install an application or add a new user

- Transformation projects require all this information and it all aligned together
Enterprise IT Transformation and why we need Discovery
Abstract Architecture for Enterprise IT Transformation

Customer and infrastructure data collection

Analytics, Design, Planning

Procurement and physical setup

Migrate and modify

Test and remediate

Synchronize and cutover

Non-customer sources

Consolidated data repositories

Migration Analytics

Project workflow and status visualization
Detailed Architecture for Enterprise IT Transformation

**Customer and infrastructure data collection** (manual and automated)
- Infrastructure data collection
- Organizational data (owners ...)
- Customer goals and plans

**Analytics, Design, Planning**
- Target options
- Technical compatibility
- Business constraints
- Placement optimization
- Wave planning
- ROI analysis

**Procurement and physical setup**
- Hardware and software ordering
- Setup of physical infrastructure if new

**Migrate, modify**
- Core migration methods:
  - Lift-and-shift
  - Copy
  - Provision and backup-restore
  - Apply changes from design
  - Start

**Test and remediate**
- Unit tests
- Comparison tests
- User acceptance tests
  If fails, back to “modify” or initial state

**Synchronize and cutover**
- Educate
- Resynchronize test system with source
- Switch operation over
- Registrations etc.

**Non-customer sources**
- External product descriptions
- Benchmarks

**Consolidated data repositories**
- Customer data
- Historic data
- General technical and financial data
  - Models

**Migration Analytics**
- Operational Improvement

**Project workflow and status visualization**

- Benchmark assessment
- Macro design
- Micro design

- Contracts

- Possibly retry

For large migrations in waves = groupings migrated at different times. Possibly even micro-design and procurement in these waves

Possibly several optimization phases
Gain vs. Pain of Transformation

Pain: We want this to be the least!

Gain: We want this to be the most!
Discovery
Customer and Infrastructure Data Collection

- This has two dimensions: What data to collect, and how. The what is determined by the repository and by the project workflow. The how must suit the data and the customer.

<table>
<thead>
<tr>
<th>Customer data collection</th>
<th>Infrastructure data collection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adapter infrastructure</td>
<td>• Standard templates for early customer overviews</td>
</tr>
<tr>
<td></td>
<td>• Standard products brought by Services team</td>
</tr>
<tr>
<td></td>
<td>• Additional scripts for data not otherwise available</td>
</tr>
<tr>
<td></td>
<td>• Preexisting 3rd-party tools at customer</td>
</tr>
<tr>
<td></td>
<td>• Including application and data dependencies</td>
</tr>
<tr>
<td></td>
<td>• Including utilization discovery</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Organizational data collection</th>
<th>Customer goals and plans</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Roles, responsibilities, meeting data etc.</td>
<td>• Timelines, budgets</td>
</tr>
<tr>
<td>• Credential acquisition data (approval processes, status of asking, finally credentials)</td>
<td>• Focus: ROI, space, power ...</td>
</tr>
<tr>
<td>• Business data for the infrastructure</td>
<td>• Constraints: facilities, vendors, consolidation layers ...</td>
</tr>
</tbody>
</table>

Data collection workflow and visualization
- What data to start / finish collecting when
- What tools to use, depending on customer inputs about goals, installed tools, approval processes
- Status tracking of credential acquisition and data collection
Manual Discovery (spreadsheets and questionnaires)

- The main discovery method in real life
- Time-consuming and error-prone but usually there is no other way to get many types of information like application owners
- Existing spreadsheets and documentation is outdated and inaccurate
  - a whole datacenter may be missing
- So typically discovery is done as:
  - get initial spreadsheets and questionnaires filled from the client
  - get reports from tools deployed by the client
  - install extra tools and get more information
  - have meetings with application owners, sysadmins, security teams, CTO to get more detailed data and validate data
  - workshops to make some common sense out of all this data
  - make common decisions and plan work
Inventory of nodes

- nobody knows about all servers, printers, routers, etc.
- need to send probing packets around to see who replies and how
- nmap is the most popular example
- based on replies can identify OS and even some software
- can trigger Intrusion Detection Systems (IDSs) and cut-off entire subnetworks from the outside networks – carefull preparation is needed
- hosts behind firewalls or most routers will not be scanned
Based on network traffic

- Modern routers have special ports to monitor traffic
- Can also insert a sniffing device
- This way it is possible to observe server-to-server traffic and look for patterns
- Can detect that say, server A is using a database on server B

- good: need no credentials on servers
- bad: need network credentials
- bad: no internal server details
Physical discovery

- Need to map physical and logical servers
- Common discovery: walk datacenters, login to each terminal, scan bar-codes
- Newer servers have RFID tags
- Some experimental discovery systems being developed
Discovery with host agents

- Internal server details can only be provided by the servers
  - Need to execute some code on the servers
- Two typical ways:
  - execute an agent software on each host and query information externally via some interface
  - login to each host using a special account and execute some code only at that time
- Discovery agents execute local commands, system calls, and read status, log, and configuration files to extract as much information about every server as they can
- Some information gathering requires root/Admin rights or per middleware accounts
Screenshot of IBM TADDM – discovery via logging in to the system and executing a Java sensor
Each per-software sensor builds a specific model (e.g., for DB2 or JFS) based on:
- configuration data
- logs
- available monitoring

Models get connected together via “URLs”
Services-friendly discovery

- For most services engagements we cannot ask for accounts on the servers before starting the real work
  - but we need data before the real work starts in order to price and plan
- Nor we can install any software
- Network probing can get you arrested in some cases
- So we have to use special tricks or do everything manually
Example Services-Friendly Discovery

- **scripts** to collect configuration, log, and connectivity data
- **parser** that processes logs and configuration files and correlates information
- **per-server data file (transparent to client)**
- *Ask system admins to execute*
- *Simple, portable, reliable*
CMDB – Change Management DataBases

- All this information is useful only if somebody can analyze it, make decisions based on it and keep it current
- IT changes over time
- BIG Wish: consolidate information into a common database and keep it current
- Problem: IT is so diverse that it is virtually impossible to model it all
  - many various different non-compatible implementations from most major enterprise software vendors
  - some IT aspects are impossible to model
  - some can be modeled in different ways
Reading List

- Nikolai Joukov, B. Pfitzmann, H. V. Ramasamy, and M. V. Devarakonda, Application-Storage Discovery, The 3rd Annual Haifa Experimental Systems Conference (Systor'10), Haifa, Israel, May 2010

- Xu Chen, Ming Zhang, Z. Morley Mao, Paramvir Bahl, Automating Network Application Dependency Discovery: Experiences, Limitations, and New Solutions, 8th USENIX Symposium on OS Design and Implementation (OSDI 2008), San Diego, CA, December 2008

Questions?