Graph-based Cloud Resource Cleanup

Netanel Cohen and Anat Bremler-Barr
The Interdisciplinary Center
Herzliya, Israel

The Problem

- As time passes, organizations that use cloud computing accumulate unused resources such as VM instances, Storage volumes and Databases. These unused resources:
  - Raise the monthly cost on public clouds.
  - Reduce capacity and degrade performance on private clouds.
  - Impose an additional operational burden.
  - Add security concerns.

General Idea

- We propose Garbo, a system that enables cloud resource cleanup by:
  1. Receiving **Core Resources** (i.e. used resources with non-cloud dependencies) as input from the user.
  2. Automatically generating a directed graph with cloud **resources** as nodes and dependency **relations** (e.g. A VM using a Storage volume) as edges.
  3. Performing Mark & Sweep on the graph, using **Core Resources** as roots.
  4. Producing a report of unused resources.

Cloud Resources Graph

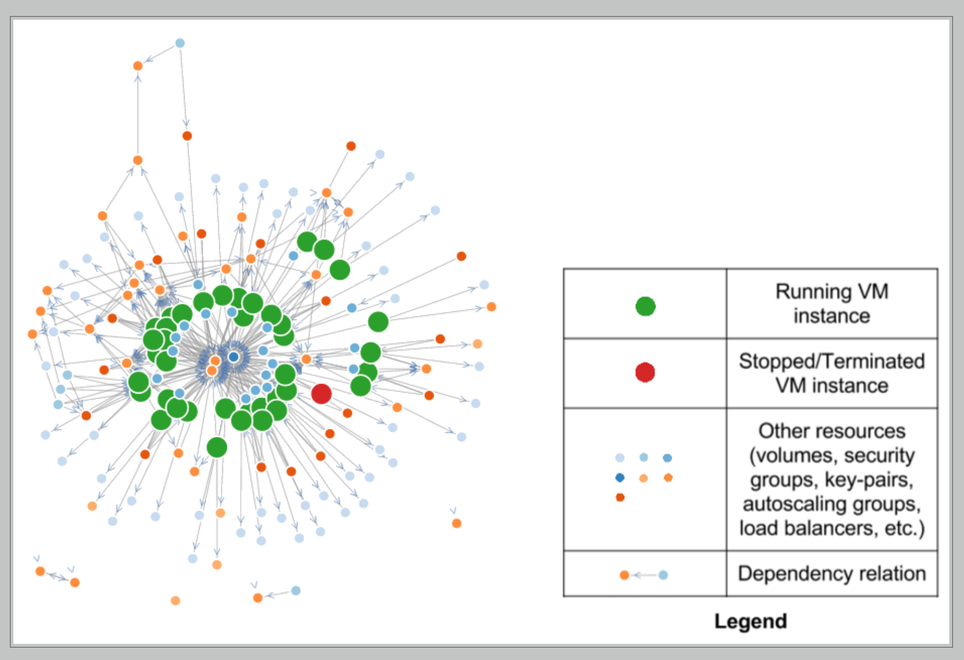

Figure 1: Cloud Resources Graph

Our Architecture

- Input of used **Core Resources**, e.g.
  - Web application's DNS record (Figure 3)
  - Batch Processing Autoscaling Group
- **Discovery Plugins** collect resources and relations from
  - Cloud API
  - Configuration Management API
  - CI/CD Tools API
- The system infers all used resources using the graph, and compiles a list of unused resources.

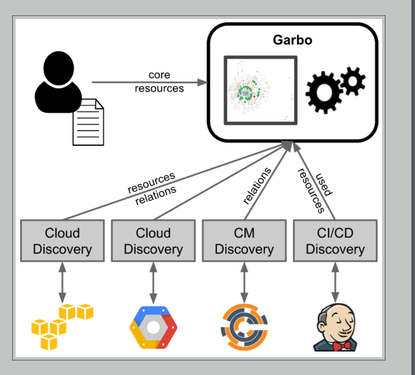

Figure 2: Architecture

Example: Core Resource in Web Application

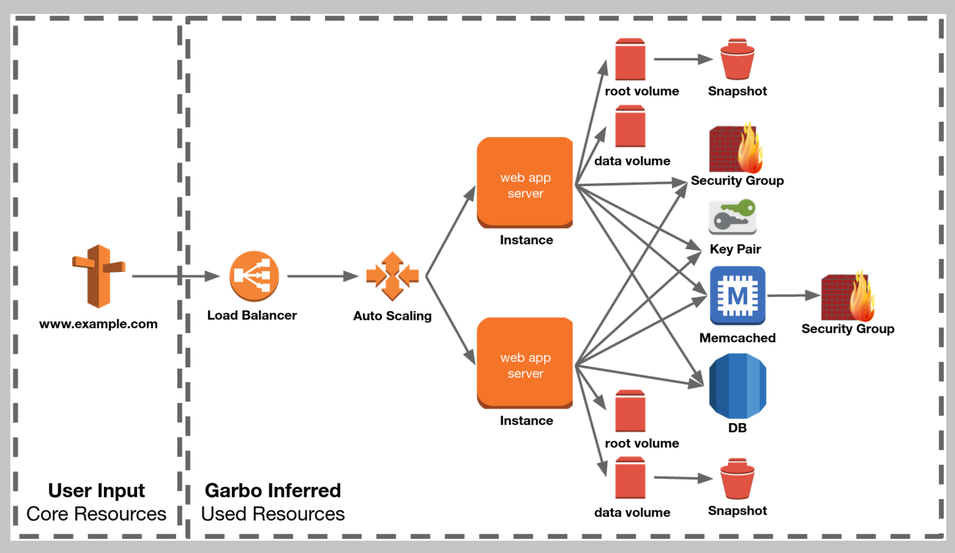

Figure 3: Core Resource in Web Application

Evaluation

- Current version implements AWS discovery
  - 11 Resource types, 18 Relation types
- Staging account of an anonymous company
  - 168 Resources, 401 Relations
  - 28 Core Resources, 8 Applications
- **Results:**
  - 14 Unused Resources (Figure 4)
  - 13 Verified by the System Administrator
  - 1 Default Cloud Resource (unused, but cannot be released)

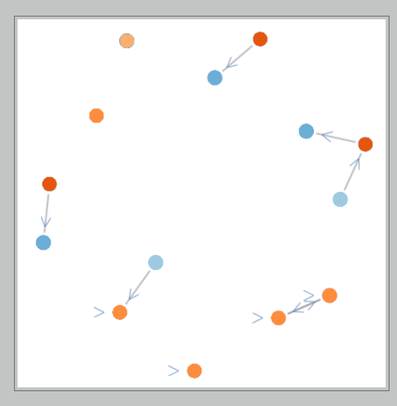

Figure 4: Unused resources output

Related Work

- Resource Cleanup
  - Poncho, Devoid et al 2013 - requires annotation per resource
  - Janitor Monkey, Netflix 2013 - requires rule set per resource type
- Other usages of a resource graph
  - Enterprise Topology Graphs, Binz et al 2012

Challenges & Future Work

- Dynamic cloud environments change rapidly
  - Asynchronous and inconsistent APIs
- Modeling resources and relations
  - Resource granularity
  - Relation directionality
- **Future Research**
  - Unique resource identification across multiple Discovery Plugins (Figure 5)
  - Detect Core Resources algorithmically
  - Online cleanup, using cloud logging (e.g. AWS Config, GCE Activity Logs)
  - Use the graph to detect failure domains

Challenge: Unique resource identification

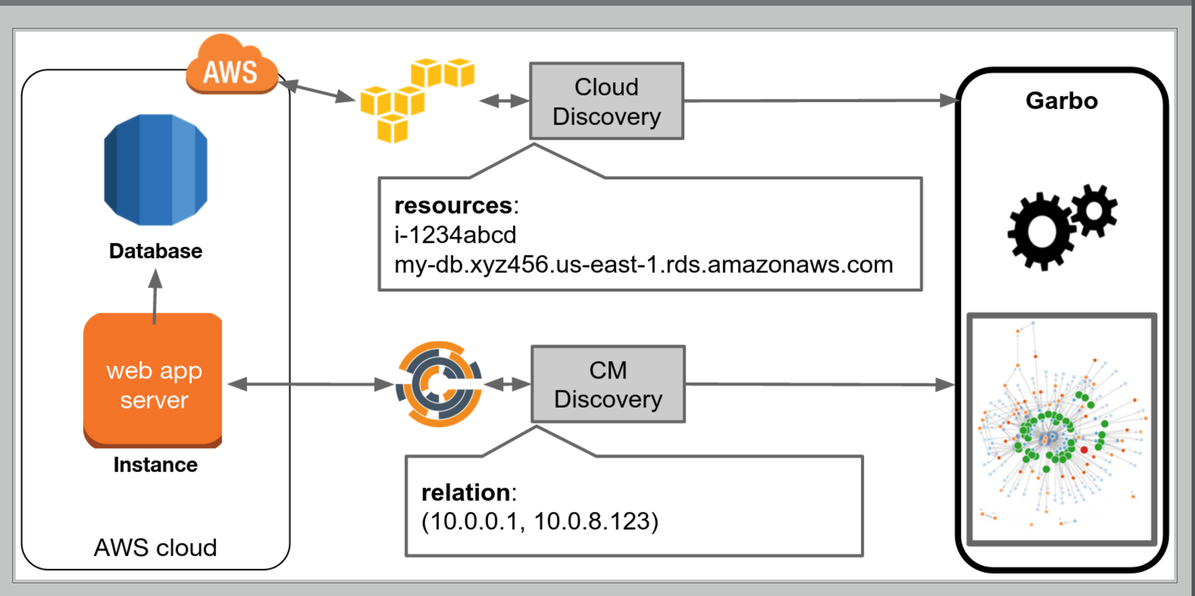

Figure 5: **Configuration Management Discovery Plugin** might identify resources using IP addresses, while **Cloud Discovery Plugin** will use cloud identifiers

Code

- Our code is available under MIT License at:
  - https://github.com/natict/garbo

Acknowledgments

- We are grateful to Avishai Ish-Shalom (Fewbytes) who provided helpful comments and suggestions regarding this work.
  This research was supported by the European Research Council under the European Unions Seventh Framework Programme (FP7/2007-2013)/ERC Grant agreement №259085.

netanel.cohentzema@post.idc.ac.il