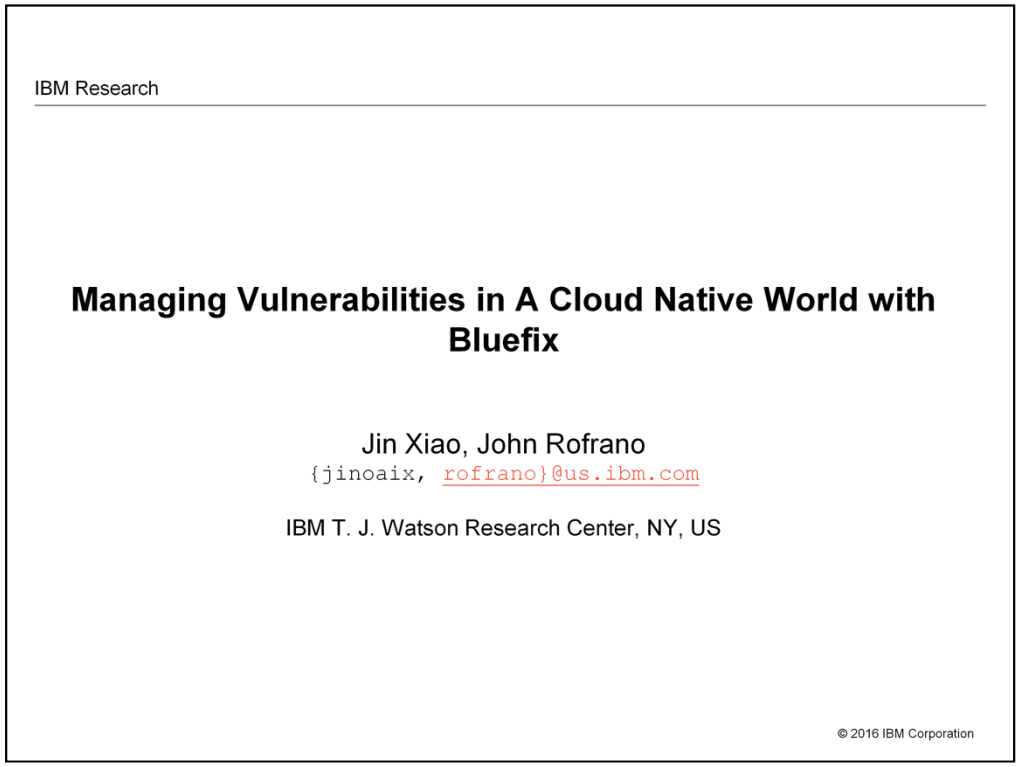


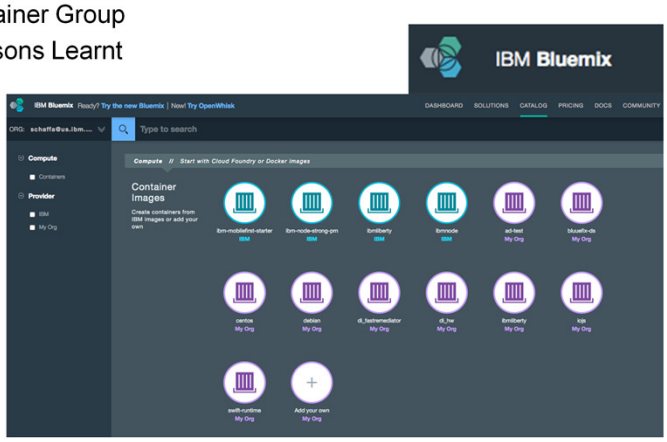
Summary

Bluefix is a solution we created for managing the vulnerabilities in a cloud native world where clients operate their own DevOp pipelines and design the same management function and assurance of that of traditional managed services. We show that the challenges imposed by container cloud and DevOp culture requires significant departure of service management design. As a case study, we examined how vulnerabilities are managed in the pre-cloud native vs. post-cloud native world, identified control points in both DevOp pipeline as well as cloud service fabric as to enable a scalable and automated vulnerability management solution for IBM container cloud. Our journey and exploration in this area has only just begun, as the lessons learned and many remaining challenges hold much promise and opportunities.

IBM Research

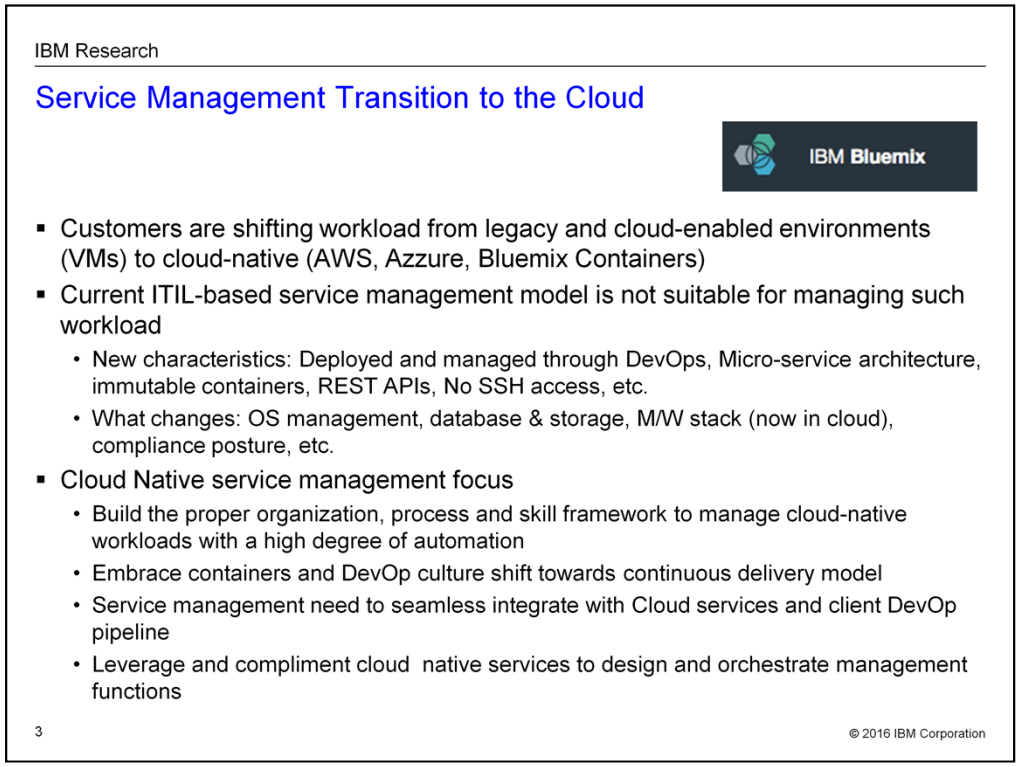
Outline

- Service Management Transition in Cloud Native World
- Challenges in Managing Vulnerabilities with Containers and DevOp
- Bluefix
- Active Deploy for Container Group
- Related Work and Lessons Learnt
- Conclusion



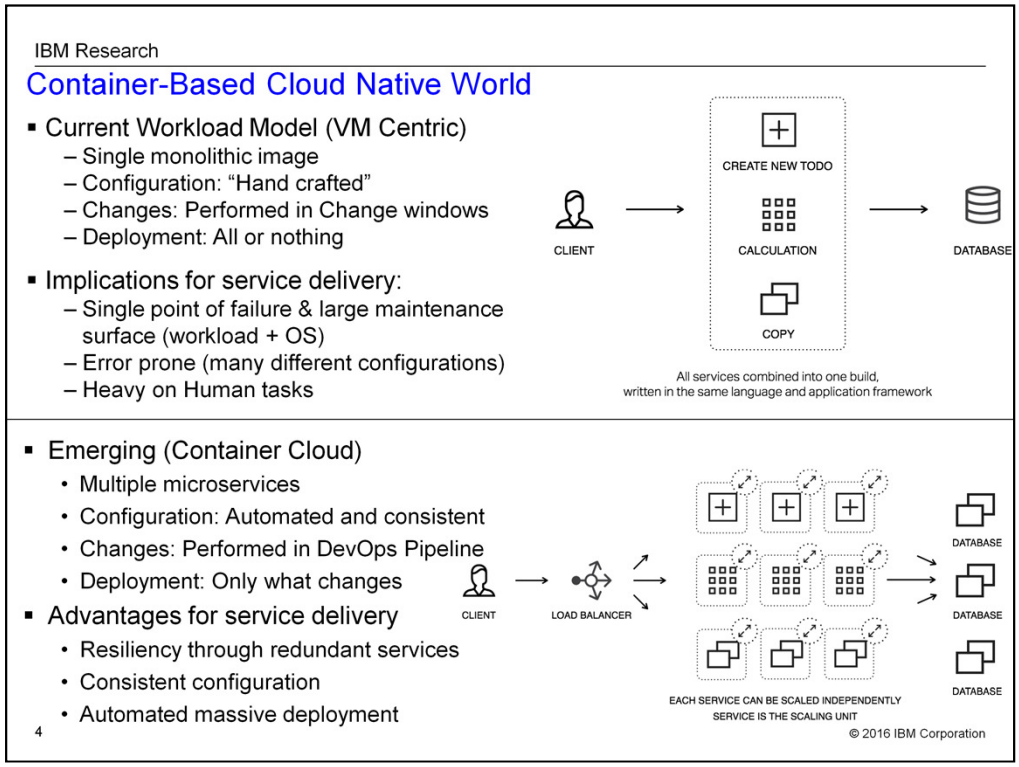
2

In this work, we first introduce the shift in service management model as enterprise clients increasingly adopt cloud native service infrastructure and DevOp culture of application development and operation. We outline the need for changing current service management model with the new challenges brought in by container-based micro-service architecture and DevOp pipelines. In particular, we contrast the way vulnerability management is performed pre-cloud native vs. post-cloud native. Then we introduce Bluefix as a solution to cloud native DevOp centric vulnerability management and reports on how the shifts in service management model are addressed in this particular problem context. We further outline the need for new technology development, the active deploy methodology for container deployment. In closing, we present related works, lessons learnt and future works.

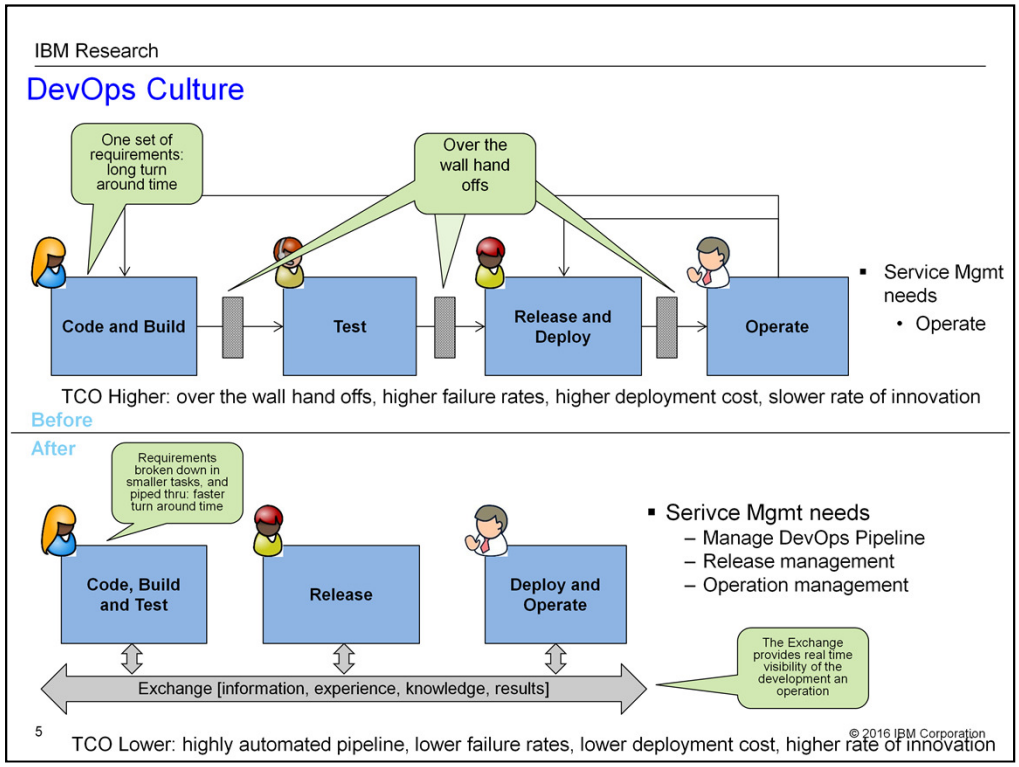


Service management is undergoing a tremendous transition today. Not only the technology landscape is shifting from physical servers to virtual machines (VMs), and now to micro-service containers, but more importantly, DevOp methodology is the new way of design, deploy and operate application and services. New services and applications are continuously introduced, composed (from reusable micro-services) and updated (often on daily basis). Meanwhile, traditional service management operates at a much slower pace with distinct service life cycle stages (design, develop, test, deploy and operate) each of which are supported by distinctive role. This model of development and management is no longer suitable as DevOps pipelines are not centrally administered and managed, containers are immutable and cannot be accessed or modified during runtime. Furthermore, many of the traditional service management tasks such as OS management, database and storage, and middleware stacks are now being managed by Cloud service providers and infrastructure providers. It begs the question whether service management as we traditionally envisioned still have a role to play in this new post-cloud native world.

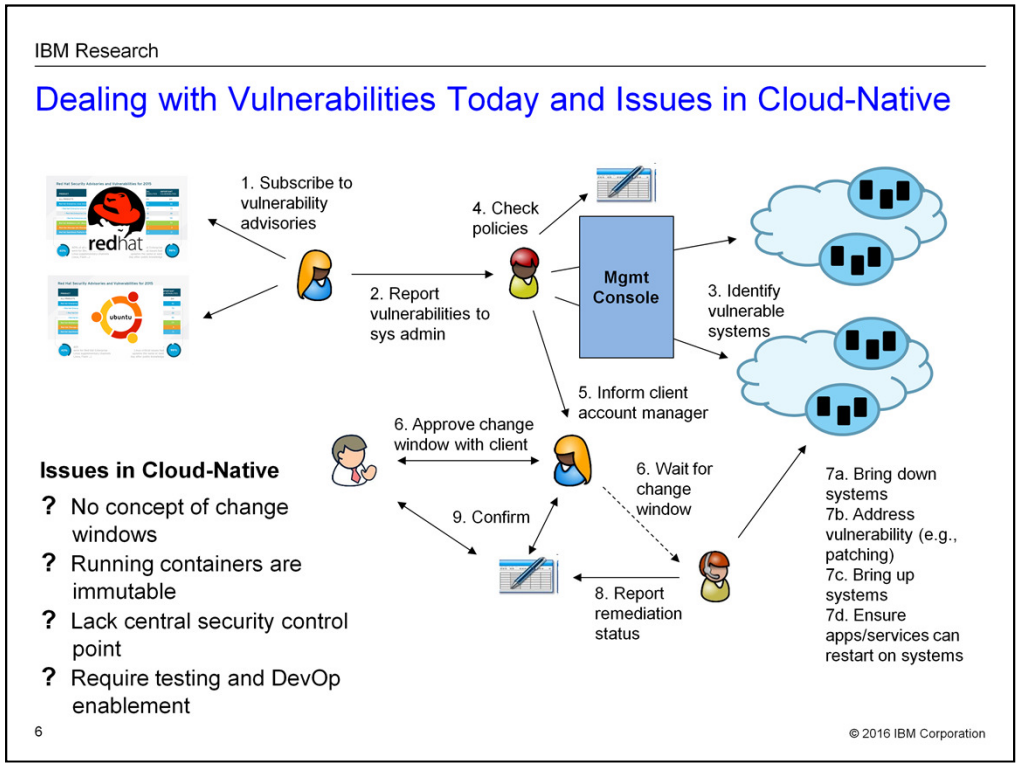
We see the need for service management shifting from traditionally dealing with managing service life cycles independent of the application design and development, and handling infrastructure centric management tasks (e.g., change, patch, identity, incident, etc.), to supporting DevOp model and pipelines with increasing automation and seamless integration.



Container-based cloud native world is disruptive to traditional service management in principle: much of traditional service management functions are designed to support the healthy operation of services run on steady state infrastructure. Incidents and security vulnerabilities are carefully identified, and system-wise addressed (e.g., SSH into the server and patch vulnerabilities, reset passwords, remove access permission violations, etc.). None of these procedures occur in container-based architecture, wherein if a service container exhibits incidents or vulnerabilities, it is destroyed and a new healthy instance is created. In fact, with the introduction of container groups, the user/administrator are not even concerned with the particularities (or physical locations of a container), as a container group manager takes care of the container life cycle management. Furthermore, any changes to be made to the container are performed as part of the DevOp pipeline, not at the steady state operational level (i.e., not touching running container instances), but rather are performed at the design and development level by the DevOp role.

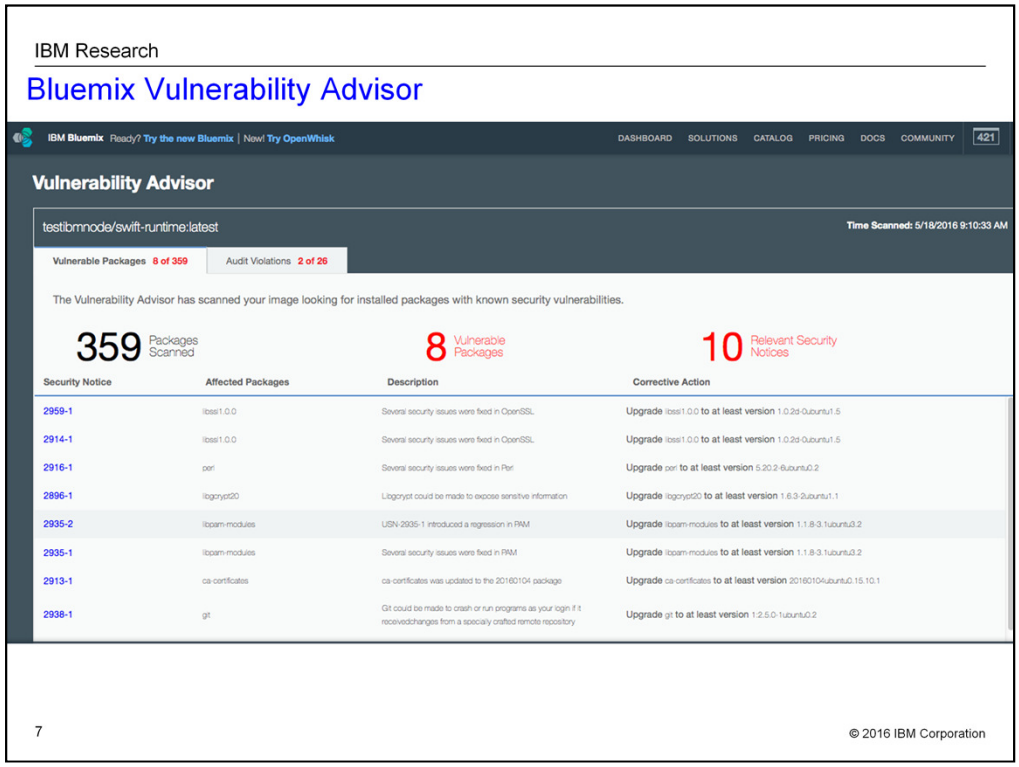


The DevOp culture drastically changes how application/service are designed, developed and operated. Traditional service management are designed to support each distinct phases of software development and each phase is conducted independent of each other, at a much longer time arch. In contrast, DevOp removes the barriers among the software development phases and requires application/service design, development, build, test, release, deploy and operation to be continuous. It also provides a full integrated pipeline that allows all of these cycles to occur in rapid succession, on daily basis, rather that weeks and months. DevOp model is highly popular in the industry today as it significantly increases the agility of business process, has rapid time-to-market, and is responsive of changes in requirements. At the same time, it invalidates traditional service management model that operates independent of the DevOp pipeline, and distinctly separates development from operations. Together with container-based architecture, DevOp requires major rethinking in service management design.



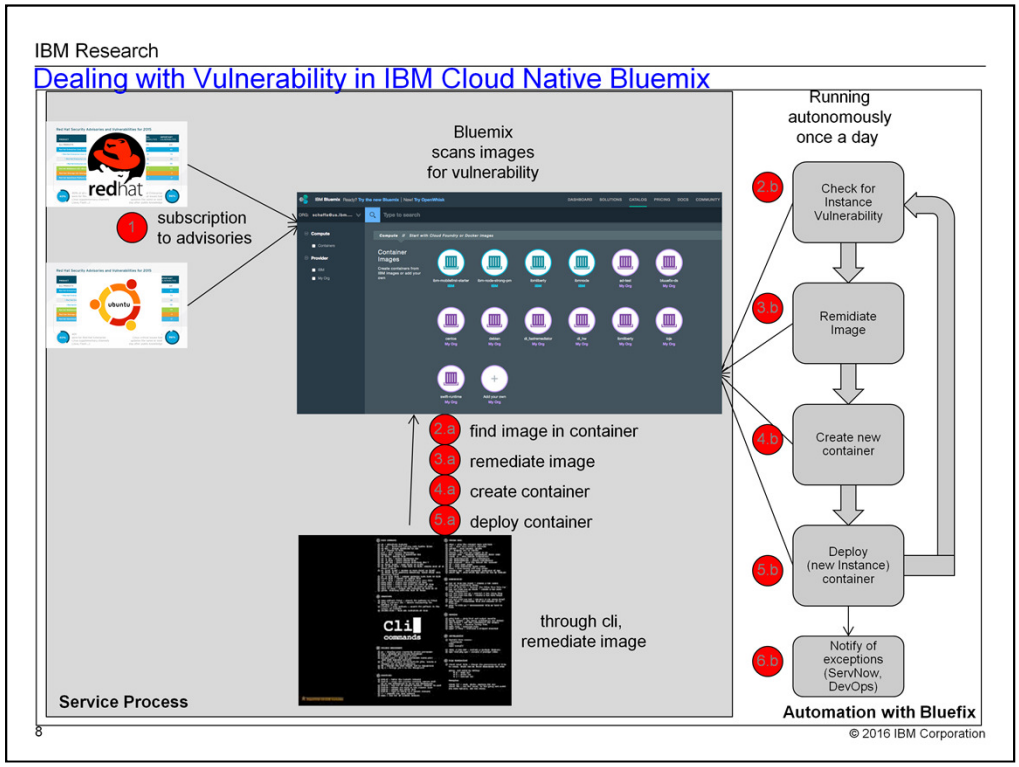
As a case study, we look at how system vulnerabilities (at OS and middleware level) are dealt with today, and outline what are the management challenges in a cloud-native DevOp world. We take vulnerability management as a case study not only because of its importance to Cloud management, but also because it has a service management pattern that are commonly found in many other management areas (e.g., change and incident management). Furthermore, system vulnerability occurring in containers are not currently dealt with by cloud service or infrastructure providers.

The diagram illustrates vulnerability management process as it applies to servers/VMs. Vulnerabilities once discovered, are identified across running systems and remediation solution is composed. This step occurs relatively quickly (hours). Then it takes time for the client to schedule a change window wherein the affected systems can be taken down and remediated. Depending on the service/workload running on the systems and the risk involved (newly updated system may cause previously running applications fail to run), the time to a change window can vary from a week to months. This is a major reason why enterprise have running systems with well-known vulnerabilities for months. In a change window, system remediation and applications restore are performed jointly by the system and application administrators. They try to verify if the application can restart and perform normally on the new system, but the verification results are rarely consistent due to the availability and variability of the application tests.

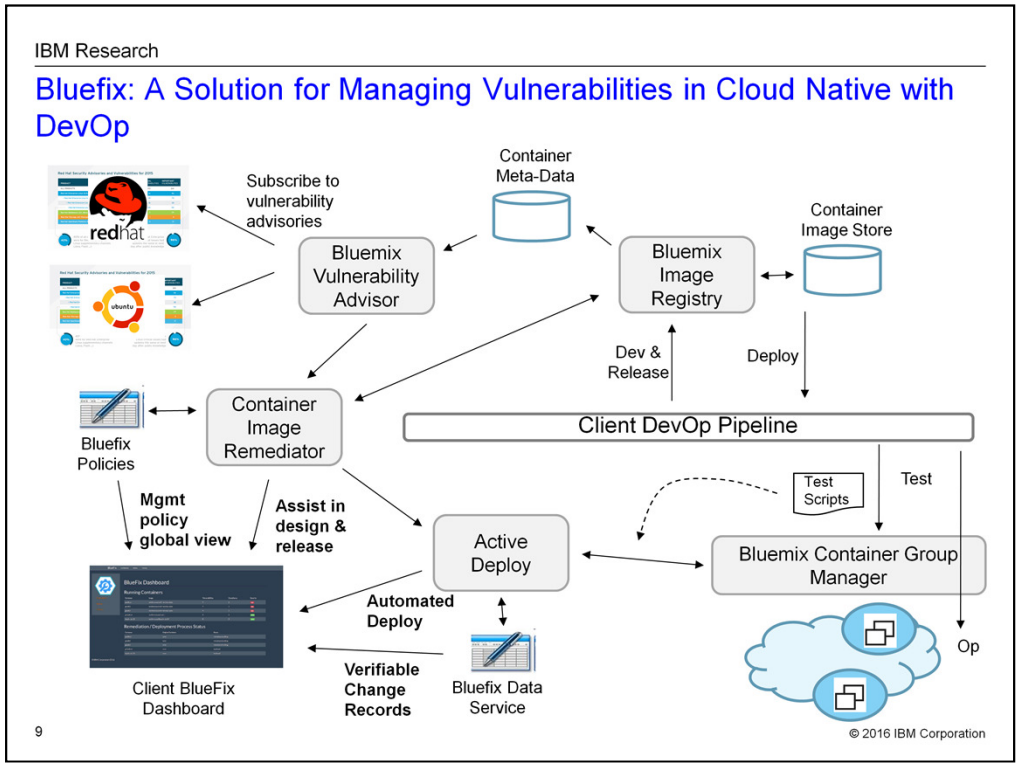


Change windows are the concept of the past. In the cloud native world, running containers are expected to be created/destroyed regularly. The container themselves are immutable and not subject to patches or changes. Any updates to the containers should be performed by changing the container images which are controlled directly by the client as part of their DevOp pipeline. This also means although vulnerabilities are regularly advertised, there isn't a central point of enforcement where by vulnerability management can be performed globally and without direct action by the clients. This makes management and automation difficult. Finally, testing and deployment configurations are specified by the clients as part of the DevOp pipeline rather than independently administered by management admin.

The above figure shows the Bluemix vulnerability advisor which provides the core functionality of advertising system vulnerabilities and reporting. On each application/service in Bluemix, it reports the specific vulnerability, remediation strategy (commonly patches) and severity level. The advisor works for containers by keeping track of container system information and dependencies as meta-data. The meta-data is generated during the container's build and release phase. This provides a key control point upon which we are able to design our vulnerability management solution.

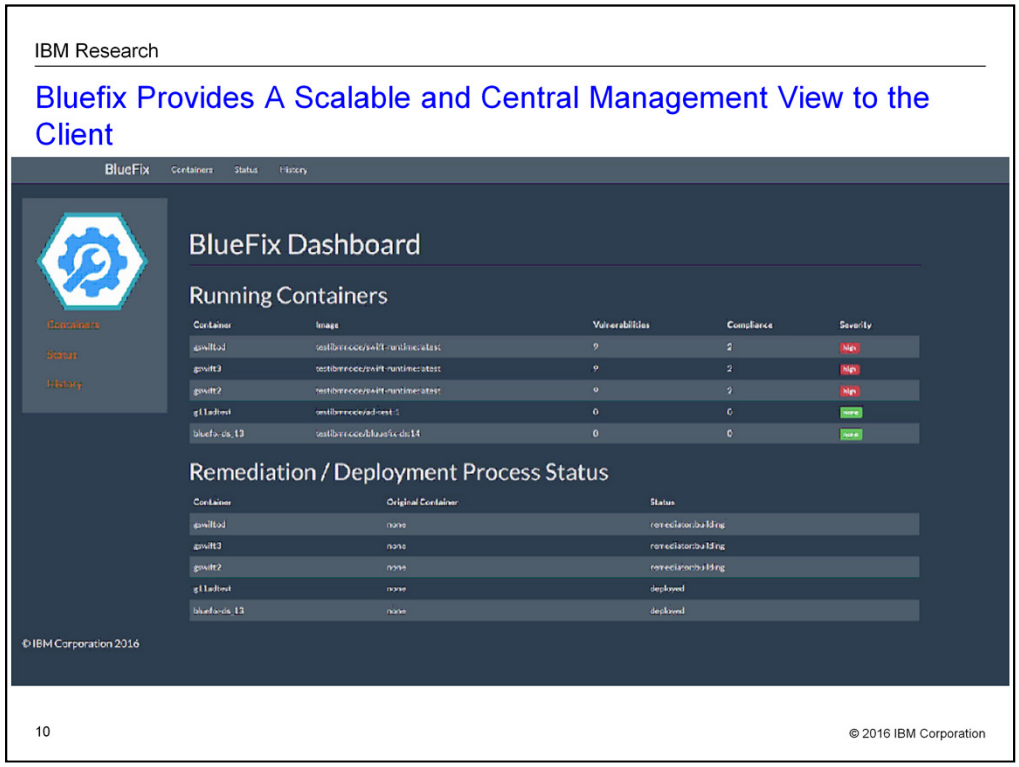


We first need to understand how system vulnerabilities are dealt with today in the cloud native world. This figure illustrates the service process involved. The aforementioned Bluemix vulnerability advisor subscribes to advisory of vulnerabilities from most major system providers (e.g., Windows, Linux, etc.). The vulnerabilities are then able to be related to particularly containers in the following way: affected containers can be identified by examining its meta-data created during the build and deploy process of the container when the container instances are created from the container images. The user/client then can remediate the container image by rebuilding the image with a remediated version of the system. This process is manually performed today. The old vulnerable containers are destroyed and new containers are created. Based on this service process, we outline a number of key steps that need to be taken if we are to automate the remediation process. We also need to identify control points along the client's DevOp pipeline that would support the automation of these steps.



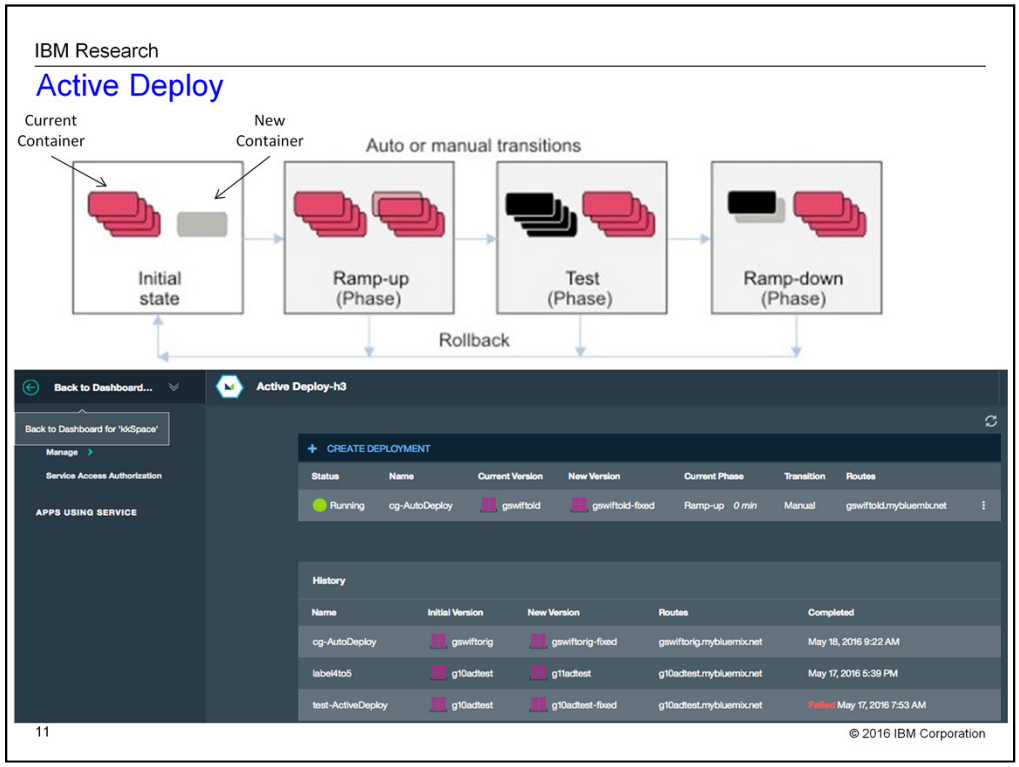
We proposed and developed Bluefix as a solution for managing vulnerabilities in cloud native world. Bluefix works seamlessly with the client's DevOp pipeline and container-based cloud infrastructure, and provides key service management features including: global management view through a single dashboard, assistance in client's application/service design and release to support continuous delivery model, automated container remediate and deployment, and verifiable change records supporting compliance audit.

Bluefix achieves these goals by identifying key DevOp pipeline control points and leveraging native container services. A client DevOp pipeline interacting with the IBM Bluemix and container cloud in the following way: as a client develops and releases application/service as a set of containers (or uses existing Bluemix services), the container image is first registered with the Bluemix image registry. This is a central point of governance in Bluemix (Amazon also has a similar registry for container cloud). A meta-data can be created for the image with the system and dependency information for the containers. The container image is then accessed by the Bluemix container group manager for deployment. The manager takes the container image and creates a number of container instances in a group based on client's specification (min., max., desired number of containers). Client test scripts are also used during the deployment to ensure the newly created containers are functional and healthy. The client's DevOp pipeline operates the container group through the manager in steady state.

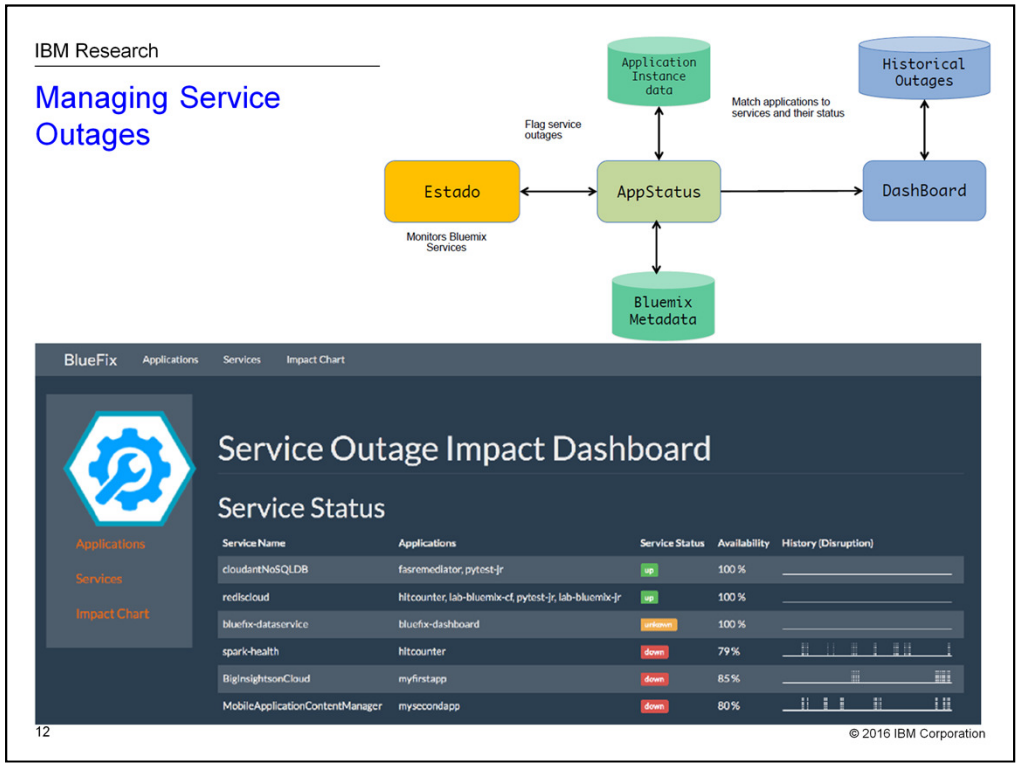


Bluefix interacts with the Bluemix vulnerability advisor and the Bluemix image registry to achieve central and automatic vulnerability discovery and identification. It takes the meta-data associated with a client's containers and matches with the system vulnerabilities to identify any existing vulnerabilities in the client's running containers. Bluefix then retrieves a copy of all of the affected containers from the container image store, and applies remediation at the system level. It does so by accessing the docker build file and modifying the system build part of the file to ensure remediation is performed. The newly updated container image is pushed back into the image store, and the client is informed. This phase takes care of patching the container image (master copy), but there are still running containers with vulnerabilities. Bluefix passes the remediated container images to the active deploy service. Client can also specify via policy which containers to auto-remediate and/or deploy with ease and flexibility. The active deploy service ensures remediated containers are built, tested and deployed. We discuss how active deploy works in the next slide.

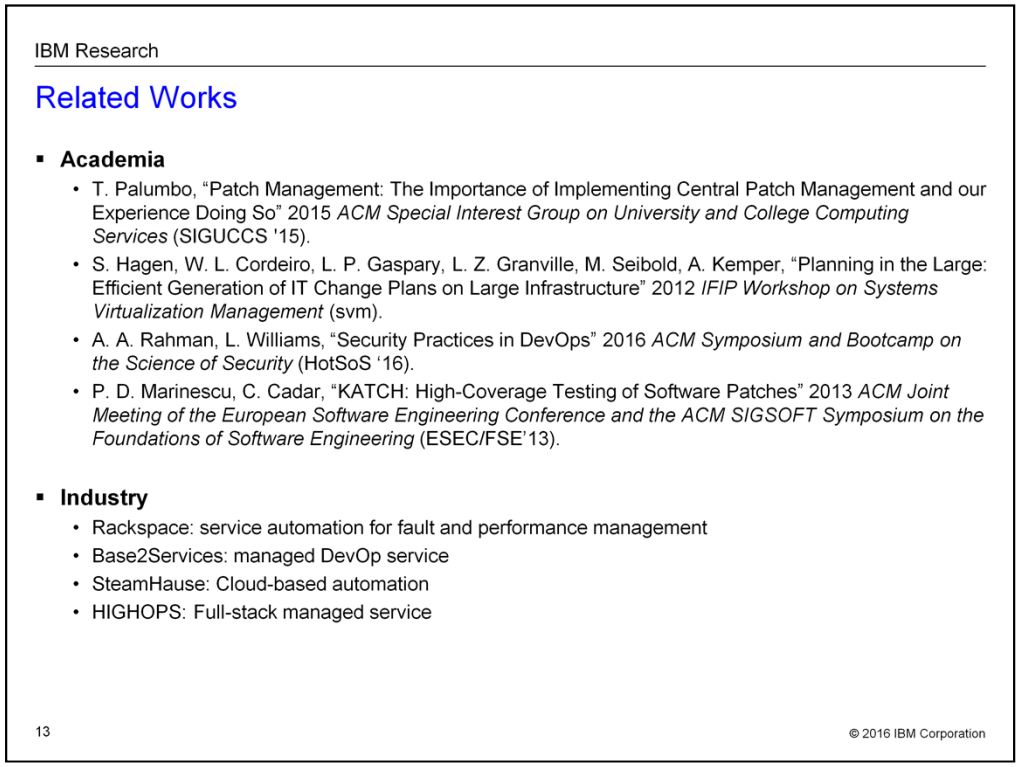
The figure above shows the Bluefix dashboard for container vulnerability management. It provides a simple global view of a client's container health. The client is able to quickly see the health condition of their containers and what remediation have taken place to address them automatically. The client can specify and manage auto-remediate and auto-deploy policies for their services.



Active deploy is our solution to update container instances in DevOp pipeline in a cloud native way. Active deploy performs live container creation, service routing and hot swapping, while performing client specific tests to ensure the new containers are deployed correctly according to specification. Active deploy works with container group manager. The figure above illustrates the different operating phases of active deploy. During initial phase, a number of current containers (i.e., with vulnerability) are running in a container group, and we desire to swap out the current containers with new containers built from new container image (i.e., remediated). During ramp-up phase, new containers are created from new image, while keeping the older containers all running. So we have two parallel groups of containers (old and new). During test phase, workloads are routed to both the new and old containers at the same time, the new containers are subjected to client's test scripts. In addition, outputs are compared between old and new containers for the same input. If everything checks out, active deploy enters ramp-down phase, during which the old containers are destroyed and all workloads are routed to the new containers. If any exception (including test failures) occurs during any of the stages, active deploy retains the old containers and removes the newly created containers. In this fashion, changes are performed on the fly with no service disruptions.



Finally, in addition to tracking the vulnerability status of application/services, we also developed solution to track the availability of Bluemix services (e.g., database service). AppStatus pulls information regarding Bluemix service availability from Bluemix's live monitoring Estado service, and correlates it to affected application/services belong to a client. It further logs historical information, such as uptime for further analysis and management optimization. In this way, Bluefix aims at providing an effective one-stop solution to security and vulnerability management for clients operating their own DevOp pipelines in a cloud native environment.



Service management in Cloud native environment has gained increased attention in the industry. For example, the listed companies provide managed service solutions with varied stages of service automation. To date, there has not been a concerted strategy and management methodology that could operate seamlessly with DevOp and container cloud.

In academia, many R&D opportunities exist in exploring and addressing the service management challenges posed in a cloud native world. Existing research in this context are still sparse with most of the patch management designs focused on traditional server or VM-based Cloud environment.

IBM Research

Lessons Learnt

- The challenges posed by DevOp and container cloud on service management requires rethinking and redesign of traditional service management model.
- To be compatible with DevOp model and to provide seamless cloud native management integration, it is important to cater the design to the specific mechanisms of the DevOp pipeline and cloud services.
- Providing agile and scalable solution to service management problems such as vulnerability management, requires systematic analysis of the process as it operates in cloud native environment, and to develop cloud native support services.

14

IBM Research

Conclusion and Future Work

- Presented Bluefix as a solution to manage vulnerabilities in cloud native DevOp environment in an automated manner.
- Explored the design and creation process needed to address service management problems in cloud native world.
- Presented challenges and opportunities for service management in cloud native world.

- Future Work
  - Adopting Bluefix to other cloud infrastructures, in order to obtain standardized management design and interoperability in hybrid cloud scenarios.
  - Expand management scope to including other aspects of security and compliance management.

15