Ágil + DevSecOps = la seguridad como código

Camil Samaha, —*Senior Manager Solutions Architecture* en Amazon Web Services (AWS), fue el invitado especial de este foro. Su charla destacó las posibilidades y beneficios del *cloud computing* en cuanto a innovación y rápido desarrollo del negocio con un alto nivel de seguridad, comparado con los entornos tradicionales.

El mayor beneficio de la nube es que se puede innovar con seguridad. En un ambiente tradicional debía sacrificarse la primera para obtener la segunda o arriesgar la seguridad para ser innovador. El *cloud computing* les permite a las empresas despreocuparse de cómo se implementará o cómo se usará una aplicación y enfocarse en el cumplimiento de la misión de sus negocios, en agregar valor a sus productos y ser competitivos en el mercado.

Camil Samaha, en su conferencia, comparó la forma de desarrollo tradicional con las ventajas que les proporciona la nube a las empresas de hoy, sean nuevas o lleven tiempo en el mercado.

Explicó que en un ambiente tradicional había que enfrentarse a dos fuerzas adversas: moverse rápido o proveer un entorno seguro, pues era muy difícil proteger el ambiente o "perímetro" de una aplicación. Lo usual ha sido colocar dispositivos de seguridad en los bordes de la red, en un acercamiento *server-centered*, con una arquitectura muy estática y rígida.

En la arquitectura del ambiente de *cloud computing* se construye con microservicios y la seguridad está incluida en cada uno de sus componentes y en cada capa de su estructura, pues se trabaja la seguridad como código. "Lo mejor es el acceso a los servicios por un 'control plan API' (*Application Programming Interface*), lo que significa que en el desarrollo de una infraestructura programable se le puede aplicar los mismos principios de ágil y de DevOps". En efecto, el desarrollo rápido y seguro implica aprovechar la metodología ágil e incorporar la seguridad en cada etapa y en cada capa de la estructura del desarrollo, lo que le permite hacerla operativa casi en el momento en que se produce. Además, por extensión, la infraestructura "sensible" se considera infraestructura como código, de tal forma que en la aplicación se integran las funciones de seguridad del sistema como código y de la infraestructura. Es lo que se conoce como DevOps, que Samaha y su equipo ponen en práctica y ofrecen en Amazon Web Services (AWS).

Foto: Natalia Fernanda Madrid Vidales

Camil Samaha, *Senior Manager Solutions Architecture* en Amazon Web Services (AWS).

AWS opera en doce regiones del mundo y ha anunciado cinco nuevas, que se desplegarán en el primer trimestre del 2017. Una región es un clúster de centros de datos, ubicados en una zona geográfica determinada; cada una se divide en "zonas de disponibilidad" conformadas por uno o más centros de datos físicos.

Cloud Computing es una oferta de servicios de TI en internet, bajo el modelo 'paga por lo que usas', que otorga un alto nivel de flexibilidad, rompe con muchas barreras impuestas en el desarrollo tradicional y facilita la entrada al mercado.

DevOps funciona con base en unos principios que hacen posible la seguridad en la nube, en donde son pieza clave la máxima automatización, el autoescalamiento y una acción proactiva y no reactiva frente a las amenazas. Estas deben detectarse en las pruebas frecuentes e implementar los correctivos en las iteraciones. Pero también es una cultura organizacional mediante la cual la empresa entiende que la seguridad depende de todos si se adueñan de ella.

Esto implica, además, disponer de los servicios de la nube para protegerla: hay más de 200 programas para la seguridad de las aplicaciones públicas. Por otra parte, la seguridad de la infraestructura no debe romper principios del *cloud computing* como el autoescalamiento, sino aprovecharlo para alcanzar las metas del negocio.

En DevOps, el desarrollo de la aplicación y su operación van juntas. En los entornos tradicionales de TI, los desarrolladores encargados de la programación y del código de la aplicación van por un lado y la operación de TI encargada de mantener el ambiente estable va por el otro; esto no permite que las organizaciones iteren rápido. Pero cuando se trabaja la seguridad como código, este concepto cobra mayor importancia,

pues significa que está presente en todas las etapas del proceso de la aplicación. Los *releases* frecuentes en un entorno ágil minimizan el riesgo, ya que los procesos de *rollback* y los cambios veloces son sencillos y se facilita identificar cualquier amenaza. El entorno DevOps permite el cambio, la toma de decisiones, las iteraciones rápidas y la innovación, uno de los principales motores de un negocio exitoso.

En el ciclo de desarrollo tradicional ágil se construyen el código y los *scripts* adicionales de prueba, todo lo cual se testea antes de liberar la versión en producción. Con el *feedback* de los usuarios se identifican las mejoras y se itera: si este ciclo termina rápido la innovación es veloz. Sin embargo, en este modelo no hay seguridad en cada paso, pues va en contra del ritmo de desarrollo y del cambio de los ciclos de iteración. Se implementa la seguridad en cada etapa con CI/CD (*Continuos Integration*, *Continuos Development*), que funciona con base en que:

- Todos los miembros del equipo son dueños de la seguridad.
- Va dirigida al cliente (*customer-driven*), es transparente (claro para todos) y hace parte de la cultura organizacional.
- *Security at scale*. El primer día no todo sale perfecto, se requiere de un acercamiento iterativo, en el que se mejora y se ajusta.
- No se es reactivo sino proactivo: se buscan las amenazas del sistema.
- DevOps brinda visibilidad de la infraestructura.

El servidor de CI y el servidor de despliegue se conectan con los diferentes ambientes DevOps en el flujo de trabajo estándar del CI/CD. En DevSecOps hay

Foto: Natalia Fernanda Madrid Vidales

Durante su presentación en el 10.º Foro de *Cloud Computing*, Camil Samaha aseguró que DevSecOps automatiza los procesos de seguridad de las aplicaciones.

seguridad en cada paso, un control de versiones y una correcta configuración del ambiente CI/CD que mantiene la integración y la habilidad de iterar rápido revisando la seguridad en cada paso.

Para que esto funcione, se deben respetar unos principios de CI/CD para DevSecOps:

1. Gran manejo de herramientas como Python, que permitan la configuración del sistema y, en última instancia, su integridad. Los *releases* nuevos se deben validar para que, por ejemplo, lo secreto se mantenga secreto.
2. Validar el código desarrollado y el código de seguridad en cada *release*. Las herramientas de trabajo y las pruebas deben garantizar un despliegue seguro.
3. La seguridad se debe tratar como si fuera el producto ofrecido al cliente. Confirmar que el esquema de seguridad está disponible para toda la organización.

Estas reglas impulsan el cambio en el flujo tradicional y el manejo del flujo de entrega continua (*Continuous Delivery*) se transforma en despliegue continuo. Los diferencia la posibilidad de automatizar el paso a producción, corroborando el código y la aplicación en cada *release* y etapa: desarrollo, pruebas y producción. Si las validaciones son buenas, se puede automatizar el proceso de modo que se desarrolle de manera rápida, con validaciones en línea.

Camil Samaha también habló de *cloudformation* que permite representar los recursos de la nube a través de archivos JSON (*JavaScript Object Notation*). Se basa en la creación de *templates* para estandarizar la configuración en un sistema. Si los archivos JSON se sacan del sistema, la aplicación dejará de correr correctamente. Se pueden gestionar diferentes versiones del archivo, mediante un controlador. Tal como en la infraestructura, se manejan como código, y se emplean los valores del desarrollo ágil.

Uno de los beneficios de c*loudformation* es que se usan herramientas con las que se trabaja cómodamente para describir y administrar el ambiente. La aplicación se vuelve escalable mediante un llamado al API que activa los servidores y bases de datos virtuales. Así se crean *templates* con la configuración especificada en las políticas establecidas en el negocio y se replican en diferentes ambientes de desarrollo u operación. Estas configuraciones ven archivos muy grandes, que por lo general se manejan en varios más pequeños, uno por función, provenientes de distintas herramientas u orígenes; todos se deben poder leer de un mismo modo —por ejemplo, mediante SSH (*Secure SHell*)—. Cada

> “En DevOps, el desarrollo de la aplicación y su operación van juntas. Cuando se trabaja la seguridad como código significa que está presente en todas las etapas”.
>
> **Camil Samaha**

microarchivo se valida antes de unirlo con los otros (*merge*), así como el resultante; cuando se despliega en el ambiente seleccionado se verifica que todo está en orden, por tercera vez. Este proceso es clave, pues garantiza que la seguridad está integrada en cada paso. Al final, se obtiene un solo archivo con todas las funciones, estandarizado con las reglas del ambiente y se integra en todos los *templates* antes de desplegarlo, de modo que se garantiza su disponibilidad en cualquier *browser*.

Es importante mantener el orden de las funciones en las validaciones: la de seguridad siempre debe ser la última. Si falla, todo el proceso lo hará. También se debe saber, en todo momento, en qué parte de la pila se está procesando y en qué ambiente o *target* y verificar que corra sobre el ambiente actual.

> “En el modelo tradicional ágil, la seguridad va en contra del ritmo de desarrollo y de los ciclos de iteración. Se puede implementar en cada etapa del proceso con CI/CD (Continuos Integration, Continuos Development)”.
>
> **Camil Samaha**

Por último, Camil Samaha dijo: “¿Por qué usar DevSecOps? Porque ayuda a determinar el nivel de cumplimiento de una aplicación. Porque es tan sencillo como crear una librería de *templates* predefinidos, porque permite determinar que se rige por los estándares deseados en cada paso del desarrollo y que la seguridad se encuentra embebida en su ADN sin restringir la capacidad de personalizar cada desarrollo”. ■

SRE permite **máxima disponibilidad** y alcance global

Este método de trabajo, en el que el desarrollo de aplicaciones es tan importante como llevarlas a producción, ha inspirado productos como Mesosphere, Prometheus, Kubernetes, PaaS como Docker, Heroku y Deis. En su conferencia, Andrés Villarroel Acosta, de Globant, explicó en qué consiste.

Foto: Julian Herzog, CC BY 4.0, http://bit.ly/2gU3UXF

Centro de Cómputo de Alto Rendimiento de Stuttgart. La tasa de servidores por ingeniero es de miles a uno en las organizaciones que trabajan con SRE.

Automatizar todo lo posible está en el corazón de la metodología de trabajo SRE. La observabilidad, el autodescubrimiento y los servicios permiten abstraer el concepto de servidor: este no es lo importante, sino los servicios prestados. Para alcanzar tal estado hay que partir de una infraestructura que proporciona funciones básicas: seguridad, almacenamiento e instrumentación (monitoreo).

La Ingeniería de Confiabilidad de Sitio (SRE, por *Site Reliability Engineering*) es una cultura disruptiva de una organización, explicó Andrés Villarroel Acosta, *Sysadmin Engineer, Analyst, Sr Adv., Cloud OPS* en la compañía de desarrollo de *software* Globant.