Patrón de portero

Proteja las aplicaciones y los servicios mediante el uso de una instancia de host dedicada que actúa como intermediario entre los clientes y la aplicación o el servicio, valida y desinfecta las solicitudes y pasa las solicitudes y los datos entre ellos. Esto puede proporcionar una capa adicional de seguridad y limitar la superficie de ataque del sistema.

Contexto y problema

Las aplicaciones exponen su funcionalidad a los clientes al aceptar y procesar solicitudes. En escenarios alojados en la nube, las aplicaciones exponen los puntos finales a los que se conectan los clientes y, por lo general, incluyen el código para manejar las solicitudes de los clientes. Este código realiza la autenticación y validación, parte o todo el procesamiento de solicitudes, y es probable que acceda al almacenamiento y otros servicios en nombre del cliente.

Si un usuario malintencionado puede comprometer el sistema y obtener acceso al entorno de alojamiento de la aplicación, los mecanismos de seguridad que utiliza, como las credenciales y las claves de almacenamiento, y los servicios y datos a los que accede, quedan expuestos. Como resultado, el usuario malintencionado puede obtener acceso sin restricciones a información confidencial y otros servicios.

Solución

Para minimizar el riesgo de que los clientes obtengan acceso a información y servicios confidenciales, desacople los hosts o las tareas que exponen los terminales públicos del código que procesa las solicitudes y accede al almacenamiento. Puede lograr esto mediante el uso de una **fachada** o una tarea dedicada que interactúa con los clientes y luego entrega la solicitud, tal vez a través de una interfaz desacoplada, a los hosts o tareas que manejarán la solicitud. La figura proporciona una descripción general de alto nivel de este patrón.

Descripción general de alto nivel de este patrón

El patrón de portero se puede usar para proteger simplemente el almacenamiento, o se puede usar como una fachada más completa para proteger todas las funciones de la aplicación. **Los factores importantes son:**

Validación controlada. El guardián valida todas las solicitudes y rechaza aquellas que no cumplen con los requisitos de validación.

Riesgo y exposición limitados. El guardián no tiene acceso a las credenciales o claves utilizadas por el host de confianza para acceder al almacenamiento y los servicios. Si el portero se ve comprometido, el atacante no tiene acceso a estas credenciales o claves.

Seguridad adecuada. El portero se ejecuta en un modo de privilegio limitado, mientras que el resto de la aplicación se ejecuta en el modo de plena confianza necesario para acceder al almacenamiento y los servicios. Si el portero se ve comprometido, no puede acceder directamente a los servicios o datos de la aplicación.

Este patrón actúa como un cortafuegos en una topografía de red típica. Permite al guardián examinar las solicitudes y tomar una decisión sobre si pasar la solicitud al host de confianza que realiza las tareas requeridas. Esta decisión generalmente requiere que el guardián valide y desinfecte el contenido de la solicitud antes de pasarlo al host de confianza.

Problemas y consideraciones

Tenga en cuenta los siguientes puntos al decidir cómo implementar este patrón:

Asegúrese de que los hosts de confianza que el controlador de acceso pase las solicitudes para exponer solo los puntos finales internos o protegidos, y que se conecten solo al controlador de acceso. Los hosts de confianza no deben exponer ningún punto final o interfaz externos.

El portero debe ejecutarse en un modo de privilegios limitados. Por lo general, esto significa ejecutar el controlador de acceso y el host de confianza en máquinas virtuales o servicios alojados separados.

El guardián no debe realizar ningún procesamiento relacionado con la aplicación o los servicios, ni acceder a ningún dato. Su función es meramente validar y desinfectar solicitudes. Es posible que los hosts de confianza necesiten realizar una validación adicional de las solicitudes, pero el controlador de acceso debe realizar la validación del núcleo.

Utilice un canal de comunicación seguro (HTTPS, SSL o TLS) entre el controlador de acceso y los hosts o tareas de confianza cuando sea posible. Sin embargo, algunos entornos de alojamiento no admiten HTTPS en puntos finales internos.

Es probable que agregar la capa adicional a la aplicación para implementar el patrón de control de acceso tenga algún impacto en el rendimiento debido al procesamiento adicional y la comunicación de red que requiere.

La instancia del controlador de acceso podría ser un único punto de falla. Para minimizar el impacto de una falla, considere implementar instancias adicionales y usar un mecanismo de ajuste de escala automático para garantizar la capacidad de mantener la disponibilidad.

Cuando usar este patrón

Este patrón es útil para:

Aplicaciones que manejan información confidencial, exponen servicios que deben tener un alto grado de protección contra ataques maliciosos o realizan operaciones de misión crítica que no deben interrumpirse.

Aplicaciones distribuidas donde sea necesario realizar la validación de solicitudes por separado de las tareas principales, o centralizar esta validación para simplificar el mantenimiento y la administración.

Ejemplo

En un escenario alojado en la nube, este patrón se puede implementar desacoplando el rol de guardián o la máquina virtual de los roles y servicios confiables en una aplicación. Para ello, utilice un punto final interno, una cola o almacenamiento como mecanismo de comunicación intermedio. La figura ilustra el uso de un punto final interno.