DJBot: Administrando las salas de PC evitando la consola

Javier Díaz, Aldo Vizcaino, Alejandro Sabolansky y Einar Lanfranco

LINTI
Laboratorio de Investigación en Nuevas Tecnologías Informáticas,
calle 50 y 120, La Plata, Argentina
{javierd,asabolansky,avizcaino,einar}@linti.unlp.edu.ar
http://www.linti.unlp.edu.ar

Resumen La necesidad de optimizar las tareas de mantenimiento y uso remoto en las salas de PC de la Facultad de Informática de la UNLP dio lugar al desarrollo de DJBot: una aplicación web realizada en Python que permite ejecutar comandos en múltiples computadoras en un solo paso. En este documento se describen los motivos y el camino recorrido para llegar a la versión de la aplicación disponible hoy en día. DJBot se encuentra liberado como Software Libre para que cualquiera que desee pueda utilizarlo y contribuir en el proyecto. Desde su desarrollo evolucionó de ser una simple herramienta ejecutable en la línea de comandos a convertirse en una plataforma de administración centralizada que simplifica el mantenimiento, actualiza los equipos y permite ejecutar tareas programadas en la interfaz web.

Palabras clave: Botnet, Django, Fabric, Python, Redis

1. El contexto

La Facultad de Informática de la Universidad Nacional de La Plata (UNLP) actualmente cuenta con tres salas de PC, las cuales suman aproximadamente 80 equipos, que habitualmente se utilizan para dictar clases de las diferentes cátedras, realizar competencias, jornadas y otras actividades extracurriculares.

Todos los equipos disponibles cuentan con dos sistemas operativos instalados: Microsoft Windows 7 y la distribución de GNU/Linux que se desarrolla en la Facultad, Lihuen GNU/Linux[2], y queda a criterio de las cátedras y de los alumnos cuál utilizar.

Las tres salas se encuentran conectadas a la red troncal de la Facultad con acceso restringido desde el exterior, pero utilizando direccionamiento IPv4 público, lo que permite que los equipos sean identificados desde Internet.

El grupo de trabajo que conforman los autores de este documento, además de encontrarse a cargo de la administración y mantenimiento de las tres salas de PC, tiene acceso a los routers y demás dispositivos de interconexión dado que también es el grupo responsable del mantenimiento del enlace troncal de datos de la Facultad.

2. La problemática

Se pueden identificar dos cuestiones principales para resolver.

Por un lado, la problemática relacionada con las cuestiones rutinarias. La gran cantidad de actividades académicas que requieren el uso diario de las salas hace que la disponibilidad de espacio temporal para realizar tareas de mantenimiento en cada computadora sea casi nulo. Peor aún es el escenario que se presenta habitualmente donde los trabajos no se hacen sobre un equipo sino que se quieren realizar tareas en todos los equipos de la sala en un solo paso. Por ejemplo, un cambio de configuración o la instalación de un software requerido por algún usuario debe realizarse sobre todos los equipos de la sala.

Junto con la problemática planteada en relación a las tareas de mantenimiento, surgió la inquietud de cómo se podrían utilizar los equipos de la sala en los tiempos en que el equipamiento está ocioso para realizar alguna tarea específica como por ejemplo una prueba distribuida de carga a un servidor web.

Hasta el día de hoy, para realizar cualquier tipo de tareas era necesario ejecutar el software en forma manual por el operador en cada una de las máquinas involucradas mediante la interacción directa con cada uno de los equipos.

Para posibilitar y facilitar la realización en tiempo y forma de estas tareas se ha desarrollado DJBot, el proyecto que aquí se presenta.

3. La propuesta

Como respuesta a las problemáticas planteadas en la sección anterior se ha desarrollado una aplicación de administración simplificada y centralizada de terminales GNU/Linux la cual se controla mediante una interfaz web.

Todo ello fue desarrollado siguiendo el principio DRY (Dont Repeat Yourself) mediante la reutilización de componentes de software libre. A modo de resumen, se puede decir que DJBot fue realizado utilizando Python como lenguaje de programación, junto con diversos componentes y bibliotecas, como el framework Django para la interfaz de usuario, la biblioteca Fabric y el protocolo SSH, entre otros.

4. El camino

En un primer intento de solución, se utilizó Parallel SSH (PSSH)[1], una herramienta que permite entre otras cosas, realizar acciones mediante la ejecución de comandos y copiar archivos en diferentes computadoras en paralelo. PSSH se encuentra en los repositorios oficiales de la distribución Debian de GNU/Linux, lo que transitivamente hace que también esté disponible en Lihuen GNU/Linux y simplifica la puesta en producción del entorno.

Por su forma de funcionamiento, esta aplicación requiere que las direcciones IP de las máquinas a las cuales les solicita acciones se encuentren listadas en un archivo de texto. Con este archivo se ejecutan conexiones por SSH a cada una de las direcciones listadas. Por los mecanismos de protección propios de SSH[4] aparecieron restricciones:

- SSH funciona utilizando clave pública-privada. Los clientes –es decir, todas las máquinas cuya IP se encuentra en el archivo– deben contar con la clave pública del que se conectará para autorizarlo en el archivo /authorized_keys. Esto se solucionó propagando la clave pública a todos los equipos de todas las salas.
- SSH mantiene un archivo de confianza (/.ssh/known_hosts) donde guarda IP y clave pública de todos los pares con los con que dialoga en algún momento. Si ante un nuevo intento conexión el par no coincide, el servidor rechaza la conexión. Históricamente las máquinas de la Facultad estuvieron configuradas mediante un servicio de asignación dinámica de direcciones, de manera que la asignación de las direcciones IP se completaba en forma aleatoria. Con el servicio de SSH en funcionamiento, en cambio, el servicio de DHCP se modificó para que cada computadora pasara a tener una única IP fija y definitiva. Así, cuando se accedió por primera vez a las computadoras mediante SSH, se pudo generar el registro de identificación de cada máquina que asocia la computadora con su IP.

Una vez concluida esta etapa, se contaba con un software funcional que cubría en forma parcial las expectativas planteadas ya que permitía la administración remota de las salas. Sin embargo, la herramienta seguía sin cumplir las expectativas en su totalidad, tanto las originales como las que fueron surgiendo a medida que el desarrollo avanzaba.

En una segunda etapa del proyecto, surgió el desarrollo de una interfaz web como respuesta a la necesidad de que las tareas de mantenimiento y soporte pudieran ser realizadas desde cualquier lugar y por personal que no necesariamente

tenga acceso a la consola de administración. El hecho de que la implementación de PSSH exija la ejecución del intérprete de comandos BASH del sistema operativo GNU/Linux sumaba direccionamientos indirectos al proceso retrasando las tareas. Por esto, se comenzaron a estudiar alternativas y apareció la biblioteca Paramiko[5] –utilizada por PSSH– que no presenta la desventaja de los direccionamientos indirectos, pero solo permite establecer las conexiones de SSH de forma simple, lo cual dificulta el envio de órdenes a los clientes.

Por ello se descartó, y buscando una alternativa se optó por Fabric[6], una biblioteca que reúne las características más útiles de PSSH y que utiliza Paramiko para realizar las conexiones SSH.

En resumen, mediante Fabric se facilita la configuración de las tareas que se realizan en los equipos de las salas, facilitando la forma de indicar en qué terminales queremos ejecutar tareas, permitiendo utilizar distintos archivos de autenticación SSH y brindando la opción de elegir entre distintos usuarios, entre otras opciones.

Como última instancia restaba decidir cómo desarrollar la parte web; hoy es muy difícil pensar en desarrollar una aplicación web utilizando el lenguaje únicamente sin utilizar algún framework de desarrollo que acorte y simplifique el proceso mediante la provision de componentes genéricos como ser filtros de seguridad, plugins de autenticación, acceso a la bases de datos o mecanismos de templates para el desarrollo de las interfaces de usuario. Si bien existen numerosos frameworks disponibles para el desarrollo, el elegido en este caso fue Django[7], uno de los frameworks más difundidos y utilizados en el mundo del software libre; Seleccionado en este caso, fundamentalmente porque tiene la particularidad al igual que la biblioteca Fabric, de estar codificado en el lenguaje de programación Python[8].

Una característica particular de DJBot es su modo de funcionamiento. Las redes de zombis, más conocidas como “botnet” en el área de la seguridad informática, son redes formadas por equipos que realizan tareas automatizadas,donde hay un controlador principal, que generalmente a través de Internet que indica qué hacer a una gran cantidad de equipos “zombis” que siguen las ordenes sin preguntar el por qué. El diseño particularmente útil y simple que permite el framework de desarrollo Django, sumado al concepto básico de las “botnets”, dieron lugar al nombre de esta aplicación: DJBot[11]. Como las partes que conforman su nombre lo indican, “DJ” hace mención a Django y “Bot” hace referencia al funcionamiento similar al de una “botnet”. Así, DJBot es un sistema que integra manejo de conexiones SSH con una interfaz web.

5. El modelo de datos

El modelo de datos representado en la Figura 1 está compuesto por cuatro entidades: Aula, Computadora, Tarea y Configuraciones.

La entidad Aula está conformada por múltiples computadoras y presenta las siguientes características principales:

1. nombre del aula,

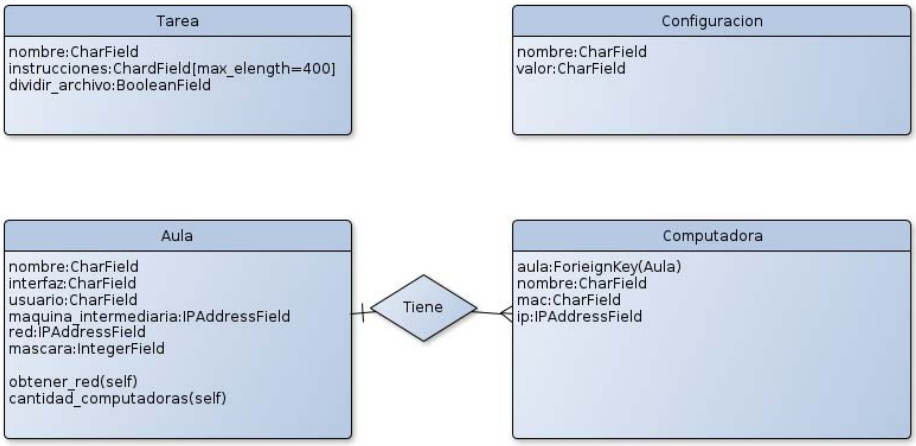

Figura 1. Clases del modelo de datos

2. usuario,
3. dirección IP de una máquina intermediaria,
4. nombre de la placa de red de la máquina intermediaria, y
5. dirección de red.

El nombre sirve para identificar el aula. La identificación mediante usuario permite realizar el inicio de sesión en cada computadora del aula. La dirección IP y el nombre de la interfaz de la máquina intermediaria, que por lo general es un router, permiten encender las computadoras de cada aula a través de la red (wake on LAN). La dirección de red, por su parte, se utiliza para asignar todas las máquinas encendidas dentro del rango de red indicado al aula específica.

La entidad Computadora incluye los valores:

1. nombre de la computadora,
2. dirección MAC,
3. dirección IP, y
4. aula a la que pertenece.

La entidad Tarea se define a través de:

1. nombre de la tarea,
2. conjunto de instrucciones, y
3. archivo opcional.

La lista de instrucciones completa la tarea en su totalidad. El archivo opcional permite compartir información entre todas las computadoras del aula, y se utiliza cuando resulta más sencillo que hacer uso de las instrucciones de comandos.

En la actualidad, la entidad Configuraciones se utiliza para definir valores predeterminados para las demás entidades. La clave SSH de las aulas, por ejemplo, está definida bajo la entidad Configuraciones. Sin embargo, en el futuro se

planea configurar claves SSH específicas para cada aula en particular. Así, la entidad Configuraciones quedaría disponible para cubrir cualquier necesidad de definición de valores predeterminados.

6. La optimización de DJBot

El desarrollo de DJBot, es decir, la integración del framework con las bibliotecas y con el modelo de datos que se describió anteriormente, se completó con el agregado de la interfaz web para lograr practicidad y flexibilidad en este trabajo.

La ejecución de tareas como la actualización del sistema y la instalación de aplicaciones nuevas, por ejemplo, implican tiempos de trabajo que al ser procesados desde la interfaz web de DJBot devolverían un error por tiempo de espera agotado. La utilización del buffer Redis[9] para almacenar las tareas solicitadas y los resultados devueltos evita, entonces, que el navegador quede como si estuviera cargando algo generándose retrasos innecesarios. Con la ayuda de la biblioteca RQWorker[10], Python se comunica con Redis y de esta manera las tareas se colocan en una cola de espera y son ejecutadas a su debido tiempo según el orden de solicitud. Esto independiza la interfaz web de DJBot para que el administrador pueda seguir interactuando con la misma sin tener demoras.

7. Los casos de uso

DJBot ya ha sido utilizado en la administración de las salas de PC de las facultad en diversas ocasiones, facilitando y automatizando las distintas tareas, entre las que podemos mencionar:

- Instalación de software solicitado por las cátedras para la realización de las actividades académicas.
- Actualización del sistema operativo ante actualizaciones de funcionalidad y seguridad de las distintas herramientas instaladas.
- Despliegue de una arquitectura para la realización de un test de stress sobre una aplicación web.

8. La liberación

DJBot está liberado como software libre bajo licencia GPL en su versión 3, una licencia que garantiza los principios del software libre permitiendo el libre uso, distribución y modificación del software a gusto de cualquiera que así lo desee.

Actualmente el código del proyecto puede encontrarse en los servidores de GitHub, uno de los repositorios de software libre más utilizados en la actualidad. Por como funciona GitHub, el lector puede descargarlo libremente desde allí; si es para usarlo puede hacerlo en forma anónima, pero en cambio si quiere generar algún aporte va a necesitar generarse un usuario en el sistema que le permita luego integrar su contribución al desarrollo.

Para acceder al desarrollo, solamente es necesario apuntar un navegador web a https://github.com/krahser/djbot.

9. El trabajo a futuro

Después de varios meses de trabajo y de tener la aplicación web funcionando, son varios los beneficios obtenidos y también son muchas las mejoras que están planificadas para ser aplicadas en el corto plazo.

Las cuestiones más importantes en las que se continuará trabajando involucran:

- Mejora de la interfaz gráfica para que resulte más simple de utilizar.
- Aplicación de mecanismos asincrónicos de notificación a través del uso de AJAX para agregar interacción a DJBot (ya se han realizado algunas pruebas con la biblioteca Dajaxice).
- Mejora de la seguridad del sistema mediante el uso de claves SSH independientes por sala.
- Agregado de un emulador de consola a la interfaz gráfica.
- Mejora del funcionamiento del buffer.
- Comprobación de la función de encendido de computadoras por red. Este ítem es necesario para evitar el tener que desplazarse hasta las salas para iniciar los equipos.
- Implementación de la función de descubrimiento de computadoras encendidas disponibles para trabajar en tiempo real.
- Carga automática de los valores de las computadoras que conforman una sala mediante una función de descubrimiento por red.

10. Las conclusiones

La disponibilidad limitada de las salas de PC de la Facultad de Informática motivó que desde el LINTI se desarrollara una aplicación web para poder realizar el trabajo de mantenimiento y administración en tiempo y forma sin interferir con la utilización académica habitual de las distintas salas de PC. Además de solucionar la problemática original, DJBot aportó soluciones colaterales, ya que no sólo permite ejecutar tareas y copiar archivos en más de una máquina al mismo tiempo, sino que ahora este trabajo no requiere de responsables expertos en la materia para poder hacerlo, ya que cualquiera puede invocar una tarea preprogramada desde la interfaz web. El acceso también es más simple; debido a que es una aplicación web, es posible acceder a la misma mediante un navegador web y así, sin más, tener acceso a todas las terminales salas de PC.

Practicidad y flexibilidad son apenas dos de los beneficios que brinda esta herramienta. Por otro lado, gracias a que ahora las computadoras de la Facultad están disponibles para ser utilizadas para realizar otras tareas en cualquier momento momento en que se encuentren ociosas y desde sitios remotos. También

se comenzaron a realizar tareas de investigación, como pruebas de estabilidad y carga distribuida de servicios web.

La práctica y experiencia de uso que se ha tenido hasta el momento son aspectos motivadores para continuar mejorando DJBot y seguir enfrentando los desafíos diarios con creatividad y precisión. DJBot es tan solo un ejemplo del amplio abanico de soluciones que están a la espera de ser creadas.

Referencias

1. Parallel SSH, http://code.google.com/p/parallel-ssh/
2. GNU/Linux Lihuen , http://www.lihuen.linti.unlp.edu.ar
3. Dinamic Host Configuration Protocol, http://linux.die.net/man/5/dhcpd.conf
4. SSH Known Hosts, http://www.linuxmanpages.com/man1/ssh.1.php
5. Paramiko, https://github.com/paramiko/paramiko
6. Fabric, http://docs.fabfile.org/en/1.6/
7. Django, versión 1.4.5, https://www.djangoproject.com/
8. Python, versión 2.7, http://www.python.org/
9. Redis, http://redis.io/
10. Django rqworker, https://github.com/ui/django-rq/
11. DJBot, https://github.com/krahser/djbot