Entrevista En Diferido 25: Eduardo Bellido

1 de Abril de 2020 a las 02:11

Comenzamos una nueva entrevista a un invitado con un perfil de administrador de sistemas e integrante del podcast Entre Dev y Ops, @entredevyops en Telegram.

Antes de todo, gracias por aceptar la invitación y apoyar a este proyecto. La primera pregunta es una presentación por parte del entrevistado, para que los lectores te puedan conocer un poco.

Entrevista En Diferido: ¿Te puedes presentar en unas líneas?

Eduardo Bellido: Hola a todos/as!

Primero de todo, agradecer la oportunidad de poder participar en el proyecto!

Y ahora las presentaciones...

Mi nombre es Eduardo Bellido Bellido (sí, repe) y mi alias en la red suele ser edubxb, y en el caso de Twitter, edu2b, todo esto y más lo podéis ver en mi (modesta/simple) web ☺ → https://edubxb.net.

El perfil/rol profesional que me define es el de Ingeniero de Sistemas o Administrador de Sistemas, que llevo ejerciendo, si no me falla la memoria, sobre unos 15 años. Actualmente presto mis servicios en Adevinta Spain (https://www.adevinta.com/es/spain/) en el cargo de SRE Lead.

A nivel personal, pues soy un apasionado del Software Libre/Open Source, e intento hacer mis pinitos y contribuir en pequeños proyectos cuando puedo. Y como ya han dicho en mi presentación, el proyecto personal que me "ocupa en la actualidad", junto a unos muy buenos amigos, es el Podcast Entre Dev y Ops (https://www.entredevyops.es).

EED: ara empezar , unas cuantas preguntas cortas para conocerte mejor a tí y a tu entorno tecnológico.

¿Qué ordenador utilizas habitualmente?
¿Sistema operativo?
¿Herramienta que más utilizas en el trabajo?
¿Conoces algún lenguaje de programación?
Un buen administrador de sistemas debe tener conocimiento en...
¿Dónde has aprendido mas?

EB: Desde que trabajo en Adevinta (de esto hace unos 5 años), y dado que las políticas de la empresa lo permiten, utilizo el equipo del trabajo también para proyectos personales, siendo ahora mismo un Dell XPS 9360, con el que estoy muy contento.

Aunque le tengo un cariño especial a mi Lenovo Thinkpad X230, que compré para uso personal hace unos años, que ya no lo uso prácticamente, pero está siempre aguardando que lo encienda y le de algo de cariño 😊.

Sistema operativo?

GNU/Linux, what else?. Concretando, Debian GNU/Linux, en una combinación de sus "sabores" Testing/Unstable/Experimental. Sí, vivo un poco al límite, pero es que soy un "ansias" de las actualizaciones...

Llevo con Debian desde la Universidad, distribución que "conocí" gracias a uno de mis primeros profesores de programación. Hice mis pinitos con RedHat antes de entrar en la universidad y use un tiempo SuSE antes de "conocer" Debian, siendo la época de transición de Potato a Woody por entonces. De esto hará ya unos 17 años... (ahora mismo me estoy sintiendo viejo 😟).

No recuerdo el año exacto, pero durante mi época de universitario hice el "switch" y dejé de usar Windows como SO principal, aunque durante un tiempo mantuve un sistema con arranque dual, cosa que dejé de hacer al cabo de unos años, haciendo uso de manera puntual de Máquinas Virtuales con Windows o Wine cuando necesitaba Windows por narices para alguna tarea.

¿Herramienta que más utilizas en el trabajo?

Esta me la sé! El terminal!

Suelo realizar la gran parte de mi trabajo en el Terminal, con Bash (soy uno nostálgico y no me he pasado a Zsh o Fish), y un sinfín de herramientas de command-line (por si resulta de interés, una gran parte de mis dotfiles están en GitHub → https://github.com/edubxb/dotfiles).

Como emulador de terminal, actualmente uso Tilix (https://gnunn1.github.io/tilix-web/), que me encanta por la integración que tiene con GNOME, que es el entorno de escritorio que uso y he usado siempre, y además tiene las funcionalidades que me gustan/necesito.

¿Conoces algún lenguaje de programación?

En mi juventud™, es decir, en la facultad principalmente, aprendí C, C++, JAVA y algo de Shell Script (para C Shell), siendo el primero el lenguaje que más me gustaba utilizar por entonces. Siempre me atrajo el low level...

En los albores de mi carrera profesional, quise aprender un lenguaje potente y versátil,

y me decidí por Python, por aquel entonces la versión sería 2.3/2.4. La alternativa era Perl en la época, pero nunca me acabó de convencer/gustar. Aunque es un lenguaje muy potente, Perl siempre me pareció muy complejo de leer, sobretodo código de otras personas.

En la actualidad, sigo con Python, que me encanta, y se podría decir que no me defiendo mal con él. Tampoco se me da mal el Shell Scripting (Bash), como a todo Administrador de Sistemas que se precie 😆.

Otros lenguajes que sigo de cerca son Go, y en mayor medida, Rust (me sigue atrayendo el low level como dije antes), el cual quiero aprender algún día de forma más o menos seria. También me resulta interesante otro no tan conocido, que es Nim (https://nim-lang.org/).

Un buen administrador de sistemas debe tener conocimiento en...

Creo que un buen Administrador de Sistemas tiene que ser capaz de desenvolverse relativamente bien (no hablo de estar a la par con un Ingeniero de Software/Desarrollador, que sería genial evidentemente) con algún que otro lenguaje de alto nivel (en mi caso es Python).

Durante años (esto con el "movimiento" DevOps™ está "mejorando", pero aún queda un buen trecho por recorrer en mi opinión), la gente "de sistemas" se ha dedicado principalmente a generar cantidades ingentes de Shell Script, lo cual, no es en sí algo malo, pero muchas tareas se hubieran abordado seguramente de forma más eficiente y mantenible con un lenguaje "con ara y ojos".

Aún así, no reniego del Shell Scripting, creo que tiene su cabida, y para ciertas tareas es mucho mejor que otras opciones.

¿Dónde has aprendido mas?

Pues como la mayoría del sector, me considero autodidacta, a base de Libros, y la cantidad de recursos que hay en Internet, se puede aprender muchísimo, así que se podría decir, que la mayoría de mi conocimiento viene de allí.

Mis años de Universidad también me aportaron mucho, y no me arrepiento para nada.

Y finalmente, en los diversos sitios en los que he trabajado, también he aprendido alguna cosa que otra, como es evidente.

EED: Participas en un podcast donde la filosofía DevOps esta muy presente. Término relativamente reciente que muchas veces no está muy claro, así que...

¿Qué es DevOps?

EB: Esta "cuestión" siempre genera, y seguramente siempre generará debate...

Está claro que aquí cada uno puede tener su opinión, pero la mía es que DevOps no es un rol, no es puesto de trabajo, es una filosofía, cultura, una serie de prácticas de como los diferentes equipos técnicos (y ojo que se podría/debería extender a no técnicos también) deberían trabajar conjuntamente o colaborar para conseguir el éxito, o ser mucho más productivos y/o eficientes en su día a día. Se traba de dejar de jugar al ratón y al gato, o al tuya mía, echándose las culpas unos a otros ante los problemas que técnicos (o de otra índole) que van saliendo como sucedía en el pasado y empezar a trabajar a buscar soluciones por y para todos. ¿Nadie recuerda a los BOFHers?

Barriendo un poco para casa, y aprovechando un artículo que escribió allá por el 2013 mi compañero de Podcast y amigo Ignasi, en nuestra web del Podcast de Entre Dev y Ops, tenéis una definición muy completa de DevOps con la que no puedo estar más de acuerdo:

https://www.entredevyops.es/posts/que-es-devops.html

Es habitual, que por estrategias de marketing/ventas, se quieran crear "conceptos" nuevos (nombres nuevos y/o molones) para poder seguir vendiendo. Esto se suele dar en empresas de servicios, principalmente de externalización de personas, o como solemos decir en el sector, "cárnicas", como una forma de dar "valor" a lo que ofrecen y diferenciarlo de lo que "vendía" anteriormente. Al final es el mismo perro con distinto collar. Está claro que no siempre es así, y hay empresas que venden/ofrecen servicios de calidad, pero muchas veces no es la norma...

Muchos son los que han caído en el error de crear un Departamento de DevOps™ o coger al actual equipo de operaciones, Administradores de Sistemas, de Red o de Bases de Datos y Ordenarlos Caballeros DevOps, y claro, todo esto sin darles ningún tipo de formación, o siquiera explicares la película... El resultado final de todo esto solo servía para cambiar la firma del correo electrónico o el perfil de LinkedIn. En resumen, era únicamente un "lavado de cara".

En el fondo con DevOps ocurre igual que con las metodologías ágiles, tiene que implicarse todo el mundo, si solo son unos cuantos, un departamento o área, los que empiezan a cambiar de "chip" pero el resto no acompaña, no se soluciona nada, y seguramente lo que genera es frustración por que lo que te habían "vendido" como la panacea, eso del DevOps, no funciona...

EED: na herramienta muy relacionada a DevOps es Docker junto con tecnologías como Kubernetes, están de moda y parece que son aplicados en todos los ámbitos. Dentro de tu ámbito laboral.

¿Se puede sobrevivir sin tener conocimientos en Docker?
¿Alguna alternativa que te guste?

EB:

¿Se puede sobrevivir sin tener conocimientos en Docker?

Sobre Docker, hay que reconocer que "revolucionó", por no decir que sentó las bases de, la contanerización como una "commodity".

Recordemos que las tecnologías en las que Docker se basa existían hace tiempo, pero las herramientas que las "explotaban" hasta la fecha (un ejemplo sería LXC → https://github.com/lxc/lxc) tenían un "punto de entrada" bastante más complejo o poco amigable, así que solo las utilizaban los/las más osados ☺. Y aquí es donde Docker entró de forma disruptiva en nuestras vidas...

Tal y como yo lo veo, creo que en el escenario actual hay que conocer Docker en mayor o menor profundidad. Es evidente que tiene enormes ventajas para muchos escenarios en los que es realmente útil, y estamos hablando desde el entorno de desarrollo (con herramientas complementarías como docker-compose) hasta llegar al entorno de producción (y aquí no podemos dejar de mencionar Kubernetes).

Si bien es cierto, muchas veces los ingenieros tenemos la tendencia de dejar atrás los "juguetes viejos" y quedarnos con los nuevos, o lo que es lo mismo, matar moscas a coñazos o pecar de hacer sobreingeniería. He visto casos de gente que se complica la vida por que docker mola o es "mainstream" (aunque ya no tanto, ahora el hype se lo queda en exclusiva Kubernetes).

Llegado aquí, decir que Docker debería formar parte del toolbox de todo buen ingeniero, sea del tipo que sea, de Sistemas, Software... ya que existen multitud de escenarios en que es la mejor opción. Pero no olvidemos que muchas veces los "clásicos", aquí podríamos incluir otras tecnologías, como p.e. máquinas Virtuales o sistemas de paquetización de toda la vida (cuando hablamos de distribución exclusivamente y no de ejecución) pueden ser más que suficientes, por no decir mejor opción.

¿Alguna alternativa que te guste?

La situación desde los orígenes de Docker ha cambiado bastante, gracias en parte o principalmente a la Cloud Native Computing Foundation (https://www.cncf.io/), el ecosistema actual de contanerización se está estandarizando permitiendo que podamos explorar opciones distintas ya que mantenemos compatibilidad y/o interoperatibilidad.

Tanto es así, que el propio Docker se convirtió en un cliente de containerd (https://containerd.io/ y https://www.docker.com/blog/what-is-containerd-runtime/) resultado de estos trabajos para estandarizar (y trabajo de "dividir" docker en componentes y ser más modular).

Aquí dejo un enlace con algunas comparativas (dejo a opinión del lector decidir qué es mejor o peor) del proyecto *rkt* (una de las alternativas) con otras opciones existentes, Docker entre ellas:

https://coreos.com/rkt/docs/latest/rkt-vs-other-projects.html

Dicho todo esto, yo de momento no he explorado en profundidad alternativas más allá de docker, así que no podría decir que haya alguna que considere mejor o me guste más.

Siendo francos, no me considero ni mucho menos un experto en la materia, así que cualquier corrección debido a un error o falta de rigor por mi marte que alguien pueda/quiera hacer será bien recibida por mi parte .

EED: Dentro de la filosofía DevOps...

¿Qué añadirías?
¿Qué eliminarías?
¿Qué modificarías?

EB: Esta pregunta es en parte complicada, dado que no hay una definición oficial, y al final todo es cuestión de interpretaciones, no tengo una referencia como la que encontraría en un diccionario o enciclopedia para poder hacer los cambios que yo creyera oportunos.

Dado que ya he expuse en una pregunta anterior cual creo que es la definición de DevOps según yo lo entiendo, dejando claro que *no es un rol*, habría que ver hasta que punto podemos definir que prácticas o no entran dentro del concepto.

Permitidme que me tome la licencia de extender mis respuestas a todo lo que envuelve al movimiento DevOps y no solo a su definición como tal.

> ¿Qué añadirías?

Quizás sería un poco más extenso en su definición para evitar palabras como:

DevSecOps
DevNetOps
Dev Whatever Ops

Al final seguimos pervirtiendo todo, es como si todo esto fuera un problema de los equipos de Operaciones, y los Devs son los buenos de la película. Como si la gente de Operaciones (de la cual me considero parte) tuviera que hacer un cambio de paradigma y el resto igual. Pues no es así y me vuelvo a repetir, esto va de que todos, repito todos, cambiemos el "chip".

La parte Dev incluye todo lo relacionado con el desarrollo como tal, y la parte Ops a su operación, buscamos cohesión y entendimiento, dejar de lado silos, no hace falta inventar más palabras para dejar claro lo que ya lo está, se puede decir más alto, sí, pero no hace falta inventar más palabras.

Para acabar, debo reconocer mi resignación, en este mundo el Marketing manda, y al final para seguir vendiendo hay que seguir "inventando", aunque sean palabras nuevas

que no signifiquen nada.

> ¿Qué eliminarías?

Eliminaría toda la jerga de "herramientas/servicios" DevOps, esto es un poco como en el Agile, por que uses Scrum no se es un empresa Agile, habría que ver como se está aplicando o si es postureo. Pues lo mismo con esas supuestas "herramientas" DevOps. Todo esto va de personas y no de "cosas", así que al final si no hay cambio de "chip" todo es fachada.

Aprovechando uno de los puntos del Agile Manifesto (https://agilemanifesto.org/):

> Individuos e interacciones sobre procesos y herramientas

Debemos mejorar en como interaccionamos entre todos, en trabajar para un mismo objetivo, no estamos compitiendo si somos de la misma empresa. Por muchas herramientas y procesos que añadamos a nuestro día a día y a nuestro flujos de trabajo, si las personas no se lo creen o no lo comparten, no hay DevOps (ni Agile) que valga...

> ¿Qué modificarías?

Aunque esto seguramente implicaría un cambio del propio nombre, creo que todo lo que pretende DevOps no está solo ligado al desarrollo y despliegue de la aplicación, hay otros muchas disciplinas que intervienen que también deberían estar presentes y entender los conceptos y las motivaciones para la existencia de DevOps como tal.

Dando ejemplos, podríamos hablar de los equipos de Diseño y UX, o incluso equipos de Marketing, o creación de contenido. Tengamos presente que en los ciclos de desarrollo de software muchas veces están presente estos perfiles ya sea en su definición o como "usuarios", y si hablamos de empresas de producto, es habitual crear software para uso interno. Así que el hecho de trabajar de una forma más cohesionada con equipos multidisciplinares para dar vida al software es una ventaja enorme.

Con DevOps se quiso romper silos entre equipos de Desarrollo y Operaciones, pero si ahora convertimos ese silo en otro más grande, todavía seguimos teniendo el mismo problema con el mundo "exterior", que en efecto, muchas veces no es técnico, pero que forma parte de un conjunto y hay que tener en cuenta.

EED: Cada vez se habla mas de la Inteligencia Artificial, su desarrollo y los múltiples ámbitos donde se está aplicando, como pueden ser ;vehículos autónomos, juegos como el ajedrez y Go, asistentes virtuales…etc

En el ámbito de los sistemas no se ve una incorporación tan grande de la IA ,aunque las herramientas están evolucionando automatizando cada vez mas tareas de administración.

¿Cuándo crees que una IA será capaz de administrar un sistema sin intervención humana?

EB: Sin duda es un campo que está avanzando de manera sorprendente, coches autónomos, asistentes como los de Google o Amazon, como bien dices, también están los "famosos" Deepfakes que dan mucho miedo...

En lo que a monitorización se refiere, ya empiezan a existir servicios que ofrecen funcionalidades de detección de anomalías que es posible que utilicen este tipo de tecnologías, más allá de eso, y dado la heterogeneidad y complejidad de los sistemas actuales, dudo que veamos algo "sorprendente" en un futuro cercano. Aunque ahora se está apostando fuerte en Kubernetes, si nos centramos en este nicho, quizás algún "grande" nos sorprenda en breve...

Por suerte o por desgracia, cada sistema es un mundo, la arquitectura, los componentes, las tecnologías utilizadas son de lo más dispares y variadas. Sistemas de IA que sean capaces de generar, o ir más a allá y solucionar problemas, lo veo complejo. Al margen de mi falta de conocimiento en este campo, no alcanzo a ver como se podría entrenar a esta IA, es decir, que datos habría que generar de incidencias reales para que empiece a ser efectiva/útil.

Por otro lado, en el escenario actual, hay decisiones a tomar que necesitan de un Humano (de mayor o menor inteligencia), además, en otras se activan protocolos de escalado, comunicaciones con proveedores o terceras partes. Así que en el caso de que empezaran a existir este tipo de IA para la operación de sistemas, me surge la duda de qué ocurriría cuando se den problemas o se tenga que interactuar con el "mundo exterior", es decir, lo que no está "controlado" por la propia IA.

A nivel de desarrollo empiezan a verse cosas interesantes. Por ejemplo, en el pasado AWS re:Invent se presentó el servicio CodeGuru (https://aws.amazon.com/codeguru/), que copiando la propia descripción de su web: Amazon CodeGuru is a machine learning service for automated code reviews and application performance recommendations.

EED: En la empresa donde trabajas, me imagino que tendrás algún poder de decisión ya sea con las herramientas que utilizas como el sistema que administras.

Imagina que tuvieras poder absoluto para diseñar el "sistema perfecto" para administrar.

¿Qué tendría, sistema operativo, políticas seguridad, herramientas de administración...etc?

EB: Partamos de que no existe el "sistema perfecto". Como dije en una de mis anteriores respuestas, para lo bueno y para lo malo, cada sistema, aunque sea funcionalmente idéntico, tendrá un diseño, y usará unas tecnologías distinas, en función de quiénes lo hayan diseñado/implementado.

Muchas veces las propias "preferencias" y/o "sesgos" nos hacen tomar decisiones poco objetivas en cuanto a las tecnologías o herramientas que usamos, somo humanos, y

eso es así.

Yo por ejemplo, soy firme creyente y seguidor de todo lo relacionado con el Software Libre/Open Source, eso ya hace que mis decisiones estén condicionadas en cierto modo a la hora de decantarme. Aunque con la edad creo que eso cada vez sucede menos ya que la madurez y/o experiencia al final juegan un buen papel a la hora de tomar decisiones.

Tenemos que tener en cuenta que a nivel empresarial el objetivo es ganar dinero, y creer o no en el Software Libre/Open Source no hará que la empresa gane más (o se gaste menos). Tenemos que tomar las decisiones en función de las necesidades y las capacidades de nuestra empresa y/o equipo. Al final como ingenieros también tenemos que tener en cuenta los costes de implantación y los operativos, no solo la funcionalidad técnica. Poniendo ejemplos, muchas veces es mejor optar por un servicio gestionado o por un SaaS, ya al realizar una evaluación de alternativas concluyamos que las horas/hombre de mantenimiento harían que una solución interna sería mucho más costosa económicamente y quizás por eso no sería asumible.

Aún así, "me mojaré" y pondré algunas de mis preferencias (por motivaciones técnicas o por mis propis sesgos) personales:

- Como stack de desarrollo seguramente iría por **Go**. Es un lenguaje muy productivo, tipado, y compilado, con lo que en función de lo que estemos diseñando, podemos tener un rendimiento óptimo, algo que sería más complicado con lenguajes interpretados como por ejemplo Python.

- A nivel de bases de datos relacional escogería **Postgresql**. Es un motor muy potente y con funcionalidaes que van más allá de lo que en definición es una base de datos relacional. Ahora mismo creo que si la compararámos con soluciones privativas, como por ejemplo Oracle, sería la única que podría plantarle cara.

- Como servidor web/balanceador, **Nginx**. He trabajado años con Apache, pero realmente Nginx es una gran aplicación, ha ganado adeptos con los años y no es para menos, es muy versátil y lo puedes hacer prácticamente todo a nivel HTTP.

- **Ansible** para orquestación/gestión de la congifuración. En su momento estuve mirando Puppet, Chef, y otra soluciones existentes, pero el modelo push de Ansible es el que más me gustó, al menos encajaba más em mi forma de hacer las cosas. Además le cogí mucho cariño al modo ad-hoc en su momento.

- Para desarrollo de tooling interno no te sabría decir, quizás también **Go**, pero **Python** ya está muy acomodado en mi toolkit personal.

- **Jenkins** probablemente para automatización de tareas y CI/CD. Es de lo más versátil que existe, hay otras soluciones, pero se quedan muchas veces cortas cuando necesitas salirte de hacer builds y pasar unit tests.

- **Git** sin dudarlo para el control de versiones, no contemplo otra opción.

- En cuanto al sistema operativo, adoro **Debian**, pero dado que **Ubuntu** tiene soporte oficial podría decantarme por este último. Aunque no tengo problemas con RedHat y derivados.

Y para no extenderme, no pondré nada más, ya que podría poner una lista interminable, pero creo que estas herramientas ya pueden dar una idea al lector de mi modus operandi.

EED: Para ser un buen administrador es necesario una serie de conocimientos, pero en un ámbito tan grande como la administración de sistemas y que puede estar relacionado con redes, virtualización,programación,seguridad,servidores...etc. Para alguién que esta empezando , no sabe que es lo primero que debe aprender para ser un buen profesional.

A continuación, pondre una lista con diversas herramientas, tecnologías,areas de conocimiento... Responde con el grado de importantancia(obligatorio,importante,opcional, no necesario...) y porque, pensando en una persona que quiere empezar en el mundo de la administración de sistemas.

Servidores web.
Chef,Puppet,Ansible y similares.
Docker y sus tecnologías asociadas.
Base de datos(SQL y NoSQL).
Programación.
Redes(hardware y software).
Auditorias de seguridad.
CMS(Wordpress u otros).
WMware.
NAS.
Servidores CI/CD.
LDAP, Active Directory y similares.
Cableado de red.

EB: Tengamos en cuenta que en cierto modo podemos acabar siento varios tipos de profesional.

Generalista

Lo que vendría siendo un maestro de todo, experto en nada, o como dicen los que practican la lengua de Shakespeare, a jack of all trades.

Aquí seríamos muy versátiles pero quizás pequemos de necesitar siempre tener apoyo de otros/as profesionales con mayor profundidad en áreas puntuales par poder abordar ciertos proyectos.

Especialista

Poco que decír aquí, nos centramos en ciertas tecnologías o incluso cierto software, pero seguramente tengamos poco espacio de maniobra profesionalmente hablando.

T-Shaped

También se habla de los perfiles T-Shaped (https://en.wikipedia.org/wiki/T-shaped_skills), que vendría a ser un generalista especializado. Vendría siendo alguien que ha adquirido un mayor nivel de conocimiento en un o varias áreas, pero que ha llegado también a adquirir, aunque en menor medida, cierto conocimiento en otras.

Yo creo que estoy entre generalista y/o t-shaped, no sé si por elección o por que mi trayectoria me ha acabado llevando hacía allí. No sabría decir.

Siempre he pensado que conocer, aunque sin ser experto, los dominios que rodean al de uno mismo proporciona perspectiva y también permite ponerse en la piel de la otra persona, para tomar decisiones mucho mejores al disponer de mayor contexto. Por eso yo en principio no soy muy fan de gente extremadamente especializada...

De la lista propuesta de conocimientos, quizás lo podríamos dividirla en varias áreas (Sistemas puros, Redes, DBA), aunque si nos enfocamos en un Adminstrador de Sistemas que es lo que a mi me define, quizás lo dejaría de la siguiente manera:

Servidores web

Imprescindible, es nuestro pan de cada día.

Chef, Puppet, Ansible y similares

La automatización debería ser el mantra de todo buen administrador de sistemas, esto no puede faltar.

Docker y sus tecnologías asociadas

Por supuesto es importante, a no ser que queramos ser un Adminstrador de Sistemas del siglo pasado.

Base de datos (SQL y NoSQL)

No es imprescindible, puede haber otros perfiles que se encarguen de esto, pero siempre es útil conocer motores de bases de datos aunque no sea en profundidad.

Programación

Sí o sí. Ya comenté que no hace falta llegar a estar a la par de un Desarrollador en una pregunta anterior, pero saber programar es algo inestimable para cualquier Administrador de Sistemas.

Redes (hardware y software)

Todo está conectado, hay que conocer como funcionan las redes y sus protocolos o estaremos perdidos... Llegar a ser un experto ya sería querer enfocarnos a ser Administrador de Redes, pero hay que tener un buen nivel aquí o tendremos problemas para diagnosticar ciertos problema que se nos puedan presentar...

Auditorias de seguridad

La seguridad es un mundo en sí, se requiere mucho tiempo para ser realmente bueno en este campo, que además requiere conocimientos tanto de sistemas como de desarrollo. Así que creo que es importante que un buen Administrador de Sistemas tenga nociones básicos de seguridad, ya que en el fondo, toma decisiones que pueden comprometer el sistema o sistemas que está administrando.

CMS (Wordpress u otros)

No considero para nada necesario, a no ser que nos vayamos a decicar por los motivos X a la implantación de CMS.

VMware

Últimamente nos movemos principalmente en entornos Cloud los que nos dedicamos a esto, pero si es un campo que nos gusta, hay trabajo y es muy interesante, pero con Docker y Kubernetes pisando fuerte, quizás nos podemos saltar este paso o no profundizar mucho en temas de virtualización.

NAS

Volvemos a lo mismo, estamosen Cloud, los conocimientos de almacenamiento serán más enfocados a las soluciones del servicio que utilices. Tener conocimientos de hardware para soluciones NAS o SAN ya queda relegado a ciertos profesionales, es decisión personal.

LDAP, Active Directory y similares

Todo depende del foco en el que te centres, yo por ejemplo estuve muy poco tiempo, apenas unos años administrando este tipo de servicios, aunque es interesante saber que existen y cual es su cometido, pero no tener por que acabar administrándolos si no es tu campo, como es mi caso.

Cableado de red

Hay profesionales especializados en cableado de red, y en la actualizad casi todo es Cloud. No creo que sea algo imprescindible en la actualidad.

Tampoco dejaría de lado otros conocimientos como Sistemas Operativos, en mi caso GNU/Linux o familia *nix (aunque hay gente que le gusta Windows 😉), protocolos de red (TCP/IP, HTTP, DNS), p.e.

Al final hay que ir centrando nuestra carrera, yo he dejado claro mi preferencia por entornos Cloud. Mi época de "jugar con hierro" la dejé atrás, pero todavía hay mucho trabajo y gente que le gusta. Así que tengamos en cuenta esto para decidir cual serán los pasos en nuestra carrera profesional, aprender todo no se puede, y las cosas se van olvidando, así que mejor mantener el foco.

Para finalizar, y esto es aplicable a cualquier profesional, no olvidemos que también hay que aprender y practicar las llamadas soft skills (https://es.wikipedia.org/wiki/Soft_skills), comunicación, empatía, etc. De poco sirve ser un crack si no se es capaz de explicar lo

que propones a los demás, o no eres capaz de "conectar" con el resto de tu equipo.

EED: Te haré una pregunta para saber si eres realmente un "influencer de la administración".

¿Qué herramienta o tecnología crees que se debe empezar a aprender para el 2020?

¿Qué herramienta o tecnología se debe empezar a olvidar para el 2020?

EB:

¿Qué herramienta o tecnología crees que se debe empezar a aprender para el 2020?

Anque no soy fan al 100%, entiendo y comprendo lo que aporta, así que quizás algo que habría que considerar serían las Service Mesh. Una de las implementaciones que más "suena" es Istio (https://istio.io/).

Las Service Mesh resuenan hace algún tiempo y hay ejemplos de empresas que ya han implantado esta tecnología, no es para todos ni para todo, pero siempre es bueno conocer que tenemos a nuestra dispoción y ver si nos encaja.

¿Qué herramienta o tecnología se debe empezar a olvidar para el 2020?

Lo que se olvida es que no es necesario o no se está utilizando. Seguramente depende el entorno dónde te encuentres tendrás nuevos retos, y se quedarán cosas atrás, no creo que haya nada que se deba olvidar motu proprio, sino que nuestra propia trayectoría nos llevará a que suceda.

EED: Participas en un podcast que se llama Entre Devs y Ops de un blog y un grupo de Telegram @entredevyops, pero me centraré en el podcast.

¿Qué puede aporta participar en un podcast a un administrador de sistema(conocimientos,proyectos, contactos...) o solo lo consideras entretenimiento?

EB: Aporta MUCHÍSIMO.

El hecho de hacer de "comunicador", que es lo que hacemos en el fondo con el Podcast, nos permite compartir aprendizajes y experiencias con los demás. Muchas veces nos obliga a "re-aprender" o "profundizar" en determinadas materias para poder

transmitir de forma clara y concisa nuestro mensaje, lo que nos enriquece y nos ayuda a ser mejores en lo que hacemos.

Por otro lado sí, en el fondo se hacen contactos, hay gente incluso que te reconoce y es gracioso cuando ocurre, además de ser muy motivador.

Evidentemente, también lo hacemos por diversión y entretenimiento. Poder dedicar tu tiempo (que es uno de nuestros activos más valiosos) a cosas que te motivan no tiene precio. Aunque a veces se hace duro y la falta de tiempo no nos permite dedicar todo el tiempo que querríamos.

EED: Como última pregunta de la entrevista, una pregunta un poco diferente.

¿Qué te hubiera gustado que te preguntase? Evidentemente , debes responder tu propia pregunta.

Pues una buena pregunta hubiera sido:

La infraestructura se está convirtiendo en una "commodity", así que parece que la figura de Administrador de Sistemas es menos necesaría.

¿Crees que el rol de Administrador de Sistemas seguirá siendo necesario?

Me encanta que me hagas esa pregunta ☺

Con los proveedores Cloud en la actualizad, y la enorme oferta que hay de servicios, PaaS, IaaS, FaaS (Serverless), etc. Sí que es cierto que mucha de las tareas que hasta hace unos años eran exclusivas de Adminstradores de Sistemas ya no son necesarias.

Pero como dicen, una imagen vale más que mil palabras, así que aquí dejo una tira cómica (de mis favoritas) que expresa mucho en cuatro viñetas:

http://www.commitstrip.com/en/2017/04/26/servers-there-are-no-servers-here/

Lo que ocurre en la actualidad es lo que pasaba hace años con los Data Centers, mucha gente contrataba servicios gestionados, que eran servidores, ahora esos servicios gestionados son los que dan los proveedores Cloud, así que los Administradores de Sistemas siguen existiendo y existirań, pero trabajan en otro sitio, en el proveedor.

Tengamos en cuenta que todo evoluciona, y lo que quizás encaje ahora más con lo que debe hacer un Adminstrador de Sistemas sería lo que se establece en el rol de SRE, Site Reliability Engineer:

https://en.wikipedia.org/wiki/Site_Reliability_Engineering

Este rol, que no es exclusivo de "gente "de Operaciones", es quién ahora se encarga que los sistemas funcionen correctamente y con garantías, pero con unas metodologías distintas, algo que encaja a la perfección con la filosofía DevOps.

Quizás el rol como tal no será el que hace unos años, pero hay que evolucionar y

adaptarse a los nuevos tiempos, trabajo hay, solo hay que reinventarse y afrontar los nuevos retos.

EED: Hoy es el último día de la entrevista y es el momento de la despedida, pero antes me gustaría agradecerte tu participación, espero que haya sido una experiencia interesante y entretenida para tí.

Puedes indicar tus métodos de contacto y si tienes algún proyecto, web, podcast o evento que quieras promocionar tienes este espacio para hacerlo.

Por último, me gustaría que me propusieras a una persona que tú creas que estaría dispuesto a participar en una futura entrevista.

Ha sido un placer , hasta la próxima.

EB: Gracias por la invitación, ha sido de lo más entretenido y he distrutado mucho respondiento las preguntas.

Si alguien está interesado en lo que hago, siempre puede pasar por mi web y allí encontrará como "localizarme" en la red:

https://edubxb.net

Para perezosos/as, me podréis encontrar aquí:

Twitter → https://twitter.com/edu2b
LinkedIn → https://www.linkedin.com/in/edubxb
GitHub → https://github.com/edubxb

Os vuelvo a dejar también la web del Podcast en el que participo, Entre Dev y Ops:

https://www.entredevyops.es

al cual os animo a suscribiros, seguro que lo encontráis interesante ☺.

Para finalizar, y barriendo un poco para casa, estoy seguro que algunos de mis compaeros del Podcast estará encatando en participar, p.e. @natx_bcn 😁.

Un saludo y gracias!