Classic System Engineering vs DevOps. Сравнение подходов к построению инфраструктуры

Uladzimir Okala-Kulak

Minsk, 2018

План

Пара слов про докладчика

- Ставил GNU/Linux на десктоп когда это ещё не было мэйнстримом (2004, Mandrake 9.1);
- Gentoo в production – нормально, если знаешь как и зачем;
- Работал с Solaris и AIX и сохранил психическое здоровье;
- Сотрудничество с Course.by, FriendlyData, ITG;
- МНС в НАН. Админ в Белгосстрах. Инженер по внедрению в ScienceVision. DevOps-инженер в ScienceSoft. Админ в SE Ranking.

Classic System Engineering

Классическая системная инженерия – дать кратко чёткое определение сложно, но все понимают о чём речь.

DevOps

- Хайповый набор практик администрирования и разработки для маленьких команд вейперов-смузисосов из blockchain-стартапов;
- Направление системной инженерии с фокусом на CI/CD, ZDD, TTM;
- «Agile» для администрирования и построения инфраструктуры, а также сопровождения команд разработки и тестирования;
- Реинкарнация ITIL и ITSM;
- Вариант культуры взаимодействия членов IT-команды;
- Решение всех проблем для небольших команд, разрабатывающих цифровые продукты;
- Родной брат подходов NoOps и SRE;
- ...
- и ещё +100500 вариантов

Для простоты будем считать...

- DevOps-инженер – системный инженер в команде, использующей DevOps подходы;
- Следование DevOps практикам влияет на выбор и направленность изменений инфраструктуры, стека технологий, архитектуры приложения.

Типичные команды и роли

System Engineering:

- QA в отдельной команде;
- Есть имплементаторы (инженеры по внедрению), релиз-менеджер, билд-инженер;
- Команда поддержки и сопровождения разбита на линии;
- В рамках одного проекта может существовать несколько относительно независимых команд разработки.

DevOps:

- Архитекторы, программисты и QA в одной команде разработки;
- 1 проект – 1 команда разработки;
- Роль имплементатора отсутствует. Позиции релиз-менеджера и билд-инженера не выделяются отдельно.

Основа инфраструктуры. Платформа

System Engineering:

- HW, Bare metal;
- Гипервизорная виртуализация и гиперконвергентные решения;
- IaaS.

DevOps:

- Гипервизорная виртуализация и гиперконвергентные решения;
- IaaS;
- PaaS;
- SaaS, FaaS.

OS

System Engineering:

- Различные UNIX-like и UNIX-base;
- Windows Server.

DevOps:

- Только Linux-base. GNU/Linux (Debian/Ubuntu, RHEL/CentOS/OEL, SLES/OpenSUSE) и BusyBox/Linux (Alpine).

ЯП для автоматизации и вспомогательного инструментария

System Engineering:

- C, C++;
- perl, ruby, php, python;
- expect, tcl/tk;
- PowerShell;
- Любой UNIX Shell (bash, zsh, ksh, fish, csh, tcsh).

DevOps:

- ruby, python;
- Golang;
- bourne-совместимый UNIX Shell (bash, zsh);
- редко JavaScript (node.js).

Характеристики скриптов

System Engineering:

- императивность.

DevOps:

- идемпотентность.

Отношение к новинкам, обновлениям, миграциям

System Engineering:

- «Работает — не трогай»;
- Много Legacy;
- Одна новая технология на один новый проект.

DevOps:

- «Тяп-ляп и в продакт» (и это нормально при правильном подходе);
- Много Bleeding edge.

Разворачивание и автоматизация инфраструктуры

System Engineering:

- Manual;
- UNIX shell, perl, python, etc;
- CMS (puppet, chef, saltstack, ansible).

DevOps:

- CMS (puppet, chef, saltstack, ansible);
- CI/CD-solutions;
- Cloud-automation tools (Terraform, CloudFormation).

Supervision and selfhealing

System Engineering:

- OS-level, service managers;
- Возможности Application servers.

DevOps:

- Встроенные средства оркестраторов и PaaS.

Scaling

System Engineering:

- Вертикальное масштабирование.

DevOps:

- Вертикальное масштабирование;
- Горизонтальное масштабирование;
- auto-scaling.

Logging

System Engineering:

- Локальное логирование;
- Ротация логов.

DevOps:

- Централизованный сбор логов;
- Логирование stateless-приложений;
- Отдельные утилиты для визуализации;
- Цепочка «коллектор – сторадж – дашборд» и ELK-подобные стэки.

Monitoring, metrics, alerting

System Engineering:

- Классические решения (типа zabbix, nagios);
- Бизнес-метрики не собираются.

DevOps:

- Цепочка «коллектор метрик–TSDB–визуализатор» + alert-manager (типичные примеры: exporters–prometheus–grafana, telegraf–influxdb–grafana);
- Monitoring as a code, alerting as a code;
- Self-monitoring application;
- При необходимости низкоуровневую инфраструктуру можно мониторить «классическим» решением, а сами приложения – современной связкой «коллектор метрик–TSDB–визуализатор».

Backup, restore

System Engineering:

- Bacula, Amanda, VEEAM и т.п.;
- «космические» решения для «кровавого энтерпрайза»;
- cron/crontab + scripts;

DevOps:

- Backup strategy as a code;
- Backup-jobs and restore-jobs on CI/CD-solution;
- Передеплой предыдущей версии вместо даунгрейда.

Debug, troubleshooting

System Engineering:

- Ручной анализ сетевых проблем (nc, telnet, tcpdump, nmap, wireshark);
- tail -f /path/to/log;
- gdb, strace, ldd.

DevOps:

- Анализ собранных логов и данных из мониторинга;
- Отказ от глубокого дебага (просто редеплой сбойного сервиса, если проблема не воспроизводится часто).

Архитектура решений

System Engineering:

- Монолит;
- SOA;
- «Трёхзвенки» и любовь к Application servers;
- Любые протоколы взаимодействия, включая бинарные.

DevOps:

- SOA;
- Microservices;
- Любовь к stateless и 12 факторам;
- REST API, JSON/XML;
- Контейнеризация.

Deploy, CI/CD

System Engineering:

- Деплой с простоем, окна обслуживания;
- Контроль версий только для кода приложений;
- Любой вариант деплоя (manual, scripts, CMS, CI/CD-solutions);
- CB и CI не различают.
- Deploy и Delivery не различают.

DevOps:

- Infrastructure as a code;
- ZDD;
- CI/CD-solution. CMS только для подготовки нод;
- Pipelines. CB-CI-CD-CT-CD;
- Учёт Vendor lock-in;
- Любовь к контейнеризации, оркестраторам, облакам;
- Учёт влияния Git Flow на Pipelines;
- Переменное число окружений;
- TTM.

Выводы, итоги, замечания

- CI/CD нужен всем;
- Стремитесь к ZDD;
- Начинать с процессов, а не с технологий;
- Регуляции мешают;
- Сложности с большими командами;
- Возможна миграция и в другую сторону (но нормальный CI/CD остаться должен);
- Конвергенция;
- Возможность отказа от пуризма и догматизма;
- Много школ и евангелистов;
- Думайте о горизонтальном масштабировании.

Спасибо за внимание.
Да будет срач в вопросах и комментах

Вопросы
Замечания
Угрозы
Оскорбления
Предложения