Установка центра сертификации на предприятии. Часть 1 / Хабр

habr.com/ru/companies/microsoft/articles/348944

Alexander Gureev February 22, 2018

Привет, Хабр! Мы начинаем новую серию статей. Она будет посвящена развертыванию службы сертификатов на предприятии на базе Windows Server 2016 с практическими примерами. Сегодня обозначим вступительные моменты и поговорим о типовых схемах развёртывания иерархии PKI: двухуровневой и многоуровневой. Обо всем этом читайте под катом.

Введение

Целевая аудитория

ИТ-администраторы, ИТ-инженеры и специалисты по безопасности, имеющие основные понятия о цифровых сертификатах.

Словарь терминов

В этой части серии использованы следующие сокращения и аббревиатуры:

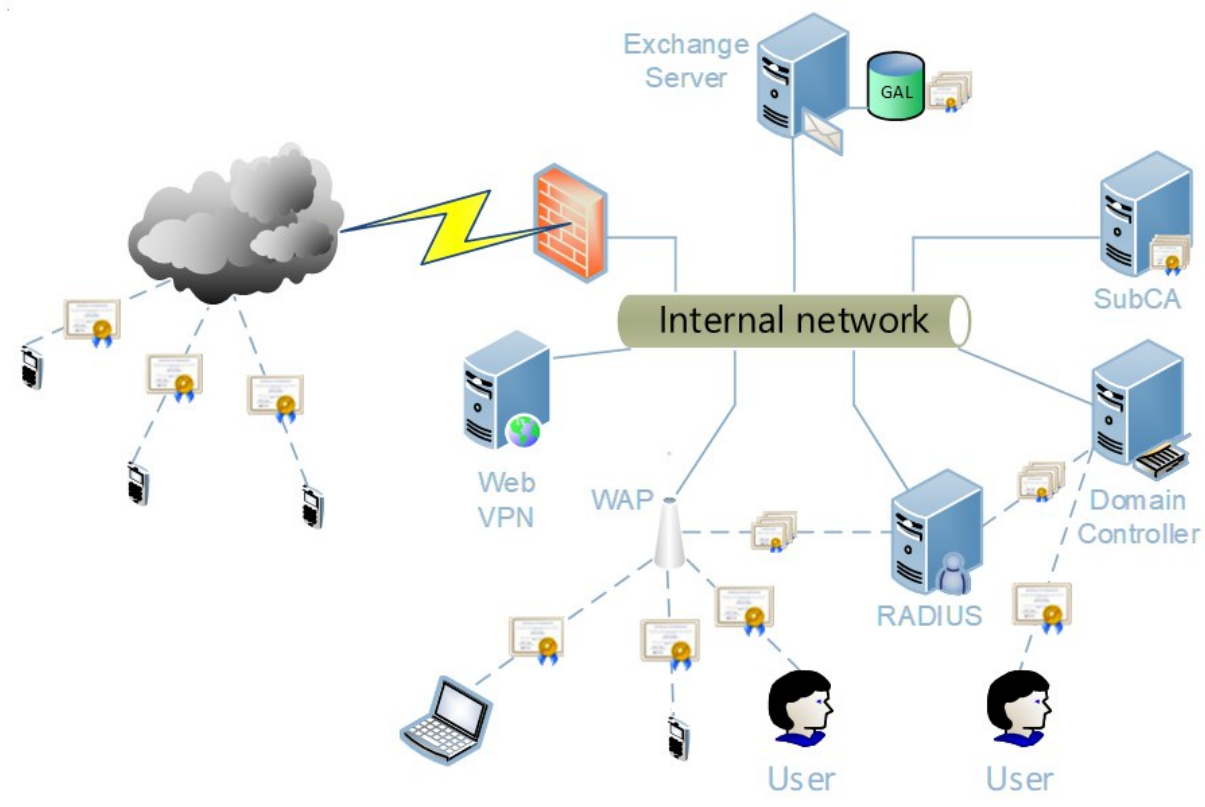
- **PKI** (*Public Key Infrastructure*) — инфраструктура открытого ключа (ИОК), набор средств (технических, материальных, людских и т. д.), распределённых служб и компонентов, в совокупности используемых для поддержки криптозадач на основе закрытого и открытого ключей. Поскольку аббревиатура ИОК не является распространённой, здесь и далее будет использоваться более знакомая англоязычная аббревиатура PKI.
- **X.509** — стандарт ITU-T для инфраструктуры открытого ключа и инфраструктуры управления привилегиями.
- **ЦС** (*Центр Сертификации*) — служба, выпускающая цифровые сертификаты. Сертификат — это электронный документ, подтверждающий принадлежность открытого ключа владельцу.

- **CRL** (*Certificate Revocation List*) — список отзыва сертификатов. Подписанный электронный документ, публикуемый ЦС и содержащий список отозванных сертификатов, действие которых прекращено по внешним причинам. Для каждого отозванного сертификата указывается его серийный номер, дата и время отзыва, а также причина отзыва (необязательно). Приложения могут использовать CRL для подтверждения того, что предъявленный сертификат является действительным и не отозван издателем.
- **SSL** (*Secure Sockets Layer*) или **TLS** (*Transport Layer Security*) — технология, обеспечивающая безопасность передачи данных между клиентом и сервером поверх открытых сетей.
- **HTTPS** (*HTTP/Secure*) — защищённый HTTP, являющийся частным случаем использования SSL.
- **Internet PKI** — набор стандартов, соглашений, процедур и практик, которые обеспечивают единый (унифицированный) механизм защиты передачи данных на основе стандарта X.509 по открытым каналам передачи данных.

Зачем нужен частный PKI?

Шифрование, цифровые подписи и сертификаты всё плотнее входят в нашу повседневную интернет-жизнь. Если об этих терминах 10 лет назад говорили мало, и далеко не всем ИТ специалистам был известен смысл этих слов, то сейчас они у многих на слуху. И этот процесс длится не год и не два, а добрый десяток лет. Сейчас мы находимся в активной фазе развития клиент-серверных веб-сервисов (привет мейнфреймам!), и значительная доля коммуникации людей и устройств перекладывается на компьютерные сети и интернет. Как следствие, новые условия диктуют новые требования к защите данных. Крупные производители ПО активно продвигают идеологию «безопасного интернета», требуя поддержки цифровых сертификатов, начиная от серверов с конфиденциальными данными, облачных служб, вплоть до кухонного чайника или бельевого утюга (IoT). Некоторые компании делают это весьма настойчиво. Так, начиная с 2017 года, компания Google считает все сайты без поддержки HTTPS небезопасными: Moving towards a more secure web. И это весьма ощутимо влияет на интернет-индустрию. Во-первых, предупреждение в браузере (Google Chrome) явно не будет радовать ваших потенциальных клиентов и просто посетителей. Во-вторых, сайты без поддержки HTTPS опускаются в поисковой выдаче. Mozilla и другие крупные вендоры также не отстают от Google: Deprecating Non-Secure HTTP. С одной стороны, компании давят на интернет-индустрию, толкая организации на дополнительные расходы, связанные с цифровыми сертификатами. С другой стороны, это — вынужденный шаг, и всем нужно идти в ногу со временем. Однако текущее положение Internet PKI не позволяет решать эти вопросы достаточно гибко и удобно, требуя существенных затрат. Например, один сертификат SSL для публичного веб-сервера вам обойдётся в сумму порядка $100, а если хотите сертификат с зелёной полосой, это вам будет

стоить ещё дороже. И это только за один сертификат! При этом автоматизация процессов находится в самом зачаточном состоянии. Для решения этих проблем крупнейшие вендоры ПО объединились и совместными усилиями наводят порядок с цифровыми сертификатами в Internet PKI. Во-первых, создан единый стандартизирующий орган — CAB Forum (CA/Browser Forum), который определяет стандартные практики для коммерческих CA, производителей веб-ПО и потребителей сертификатов. Во-вторых, активное продвижение некоммерческого CA (но глобально доверенного) Let’s Encrypt для обеспечения бесплатными сертификатами с возможностью автоматического продления. Казалось бы, это решает все проблемы безопасной коммуникации, и частные PKI (разворачиваемые в пределах организации) стали сразу ненужными. В какой-то, да, если частный PKI занимался обслуживанием внешних серверов (веб, VPN и т.д.). Но сервисы наподобие Let’s Encrypt в настоящее время покрывают лишь узкий спектр корпоративных потребностей в сертификатах. Например, не покрываются сертификаты для шифрования документов, почты, цифровых подписей. Часть задач, как аутентификация клиентов не покрывается совсем. Ещё одним ограничением является то, что для использования публичных сертификатов, выданных коммерческими CA вам необходимо иметь публичный домен. Получить сертификат для веб-сервера на имя domain.local от Let’s Encrypt технически невозможно. Именно поэтому актуальность частных PKI остаётся на очень высоком уровне. Пример использования частных PKI в корпоративной среде изображён на следующем рисунке:



Если компания решает использовать частный PKI в пределах организации, встаёт другой насущный вопрос: как же правильно его организовать, чтобы он отвечал

современным практикам и стандартам хотя бы в пределах организации. В интернете можно найти многочисленные статьи о том, как развернуть PKI в компании на основе Active Directory Certificate Services (ADCS). Но в большинстве своём они изобилуют ошибками, исходят из неверных предпосылок и нередко являются копированием какого-то исходного (не всегда удачного) материала, а к имеющимся фундаментальным ошибкам привносят ещё и свои. Как следствие, многочисленные провалы в развёртывании PKI. Об этом можно судить по количеству соответствующих тем на форумах (в частности, TechNet Server Security). Качественной документации на английском языке не хватает, а на русском… Этот цикл статей призван восполнить этот пробел и систематизировать современные наработки.

Общие положения

PKI является технологией безопасности, которая основывается на стандарте **X.509** и использует цифровые сертификаты. Целью PKI является повышение безопасности ИТ инфраструктуры предприятия за счёт предоставления механизмов по защите данных от несанкционированного доступа и незаконного изменения данных (подделка данных). Это достигается двумя основными криптографическими механизмами:

- **Шифрование** – защищает данные от несанкционированного доступа третьих лиц путём шифрования данных криптографическими ключами. Только пользователи, имеющие необходимые ключи, могут получить доступ к данным. Шифрование обеспечивает секретность данных, но не защищает от их подмены.
- **Цифровая подпись** – защищает данные от несанкционированного изменения или подделки путём применения к данным специальных алгоритмов, которые образуют цифровую подпись. Любые манипуляции по изменению данных будут немедленно обнаружены при проверке цифровой подписи. Цифровая подпись обеспечивает не конфиденциальность данных, а их целостность. Путём комбинирования шифрования и цифровой подписи можно организовать обеспечение конфиденциальности и защиты данных от несанкционированных изменений.

Типичная инфраструктура PKI состоит из следующих компонентов:

- **Центр Сертификации** (ЦС) – служба, предоставляющая цифровые сертификаты потребителям и обеспечивающая функционирование PKI.
- **Сервер отзыва** – служба, предоставляющая информацию о списках отозванных (скомпрометированных или недействительных) сертификатов, выпущенных конкретным ЦС.
- **Клиент** – получатель заверенного цифрового сертификата от центра сертификации. Клиентами могут выступать люди, устройства, программное обеспечение, а также другие ЦС.

Структура ЦС и сертификатов выстраиваются в древовидную иерархию. Каждый ЦС может выполнять одну или несколько (совмещать) ролей:

- **Корневой ЦС** – специальный тип ЦС, который имеет самоподписанный сертификат и является корнем дерева (отсюда и название). Этот тип ЦС является стартовой точкой доверия ко всем сертификатам в данной иерархии (дерева). Иными словами, клиент должен явно доверять конкретному корневому сертификату (а именно, комбинации: издатель и открытый ключ), чтобы доверять сертификатам, находящимся в остальной части дерева. Важно отметить, что доверие транзитивно. Клиент при проверке конечного сертификата будет выстраивать цепочку (путь) от конечного сертификата до вершины иерархии (корневого сертификата). И если клиент доверяет вершине, то будут и основания доверять конечному сертификату на правах транзитивности.
- **ЦС политик** – технически, это такой же ЦС, как и все остальные (в разрезе иерархии), с тем отличием, что дополняется внешними политиками и ограничениями по выдаче и использования цифровых сертификатов.
- **Издающий ЦС** – это ЦС общего назначения, который выполняет подпись и выдачу цифровых сертификатов потребителям.

Для понимания процесса построения цепочек сертификатов рекомендую прочитать следующую статью: Certificate Chaining Engine — how it works. Данная статья ориентирована на платформу Microsoft CryptoAPI, она также справедлива (за некоторыми исключениями) и для других реализаций криптографических платформ. Поскольку ЦС выстраиваются в древовидную иерархию, возможно организовать многоуровневую иерархию, где на каждом уровне ЦС будет выполнять как роль издающего ЦС, так и дополнительные функции. В самом простом случае один ЦС может совмещать все роли, т.е. быть корневым, обеспечивать какие-то политики выдачи и выдавать сертификаты конечным потребителям. Более крупные компании и/или с более зрелой организацией ИТ-процессов уже используют разделение ЦС по ролям. Например, в головном офисе держат корневой ЦС, выдающий сертификаты только другим ЦС, которые уже на себе накладывают политики выдачи. Они могут не обслуживать напрямую конечных потребителей, а выдавать сертификаты другим подчинённым ЦС, которые, в свою очередь, и будут обслуживать конечных потребителей. В каждом подходе есть свои плюсы и минусы, которые будут рассмотрены ниже.

Отзыв сертификатов

Помимо задач по выпуску сертификатов, каждый ЦС периодически выпускает списки отзыва (Certificate Revocation List, CRL). Как и сертификаты, целостность списков отзыва обеспечивается цифровой подписью. CRL содержит серийные номера сертификатов, действие которых прекращено по какой-либо причине до официального истечения срока действия сертификата. Таким образом ЦС

обеспечивает своевременное изъятие недействительного сертификата из оборота. Каждый клиент после установки доверия сертификата через цепочку должен убедиться, что ни один сертификат в цепочке не был отозван своим издателем. Для этого клиент перебирает каждый сертификат в цепочке, выбирает CRL предоставленный издателем и проверяет наличие/отсутствие текущего сертификата в списке CRL. Если текущий сертификат находится в CRL, то доверие к сертификату (и всем ветвям дерева под ним) автоматически обрывается. Фактически, если корневой ЦС отозвал все свои непосредственно изданные сертификаты, то ни один сертификат под этим корнем не будет доверенным вне зависимости от высоты иерархии. Здесь следует отметить один крайне важный и принципиальный момент: **невозможно отозвать корневой (самоподписанный) сертификат**. Т.е. если по какой-то причине он был скомпрометирован, его можно отозвать только принудительным удалением сертификата из хранилища сертификатов каждого клиента. Дело в том, что ЦС не определяет списки отзывов для самого себя, это делает издатель. В случае самоподписанного сертификата, ЦС является издателем самого себя. И при попытке включить себя в свой же список отзыва получается неопределённость: сертификат ЦС включен в CRL, который подписан ключом этого же ЦС. Если предположить, что сертификат ЦС недействителен, то и цифровая подпись на CRL является недействительной. Как следствие, невозможно достоверно утверждать, что сертификат корневого ЦС отозван. Причём, отзыв корневых сертификатов не предусмотрен и основным регламентирующим стандартом, RFC 5280, параграф §6.1 которого гласит:

> When the trust anchor is provided in the form of a self-signed certificate, this self-signed certificate is not included as part of the prospective certification path.

А на отзыв проверяется только проспективная цепочка. Если говорить в контексте Microsoft ADCS, то тут ситуация усугубляется ещё больше. В частности, вы технически можете отозвать корневой сертификат при помощи certutil.exe или CryptoAPI. Но как только вы это сделаете, ЦС не сможет подписать ни один CRL, как следствие, сертификат ЦС никогда не попадёт в CRL. Более того, даже если использовать различные утилиты (тот же certutil.exe), можно насильно включить сертификат ЦС в CRL, но это мало чем поможет. Дело в том, что конфигурация по умолчанию CryptoAPI даже не пытается проверить корневой сертификат на отзыв. Проблема отзыва корневых сертификатов во многих случаях будет одним из решающих факторов в выборе подходящей для компании иерархии ЦС. А также будет диктовать дополнительные меры безопасности для корневых ЦС, чтобы предельно снизить риск компрометации корневого ЦС.

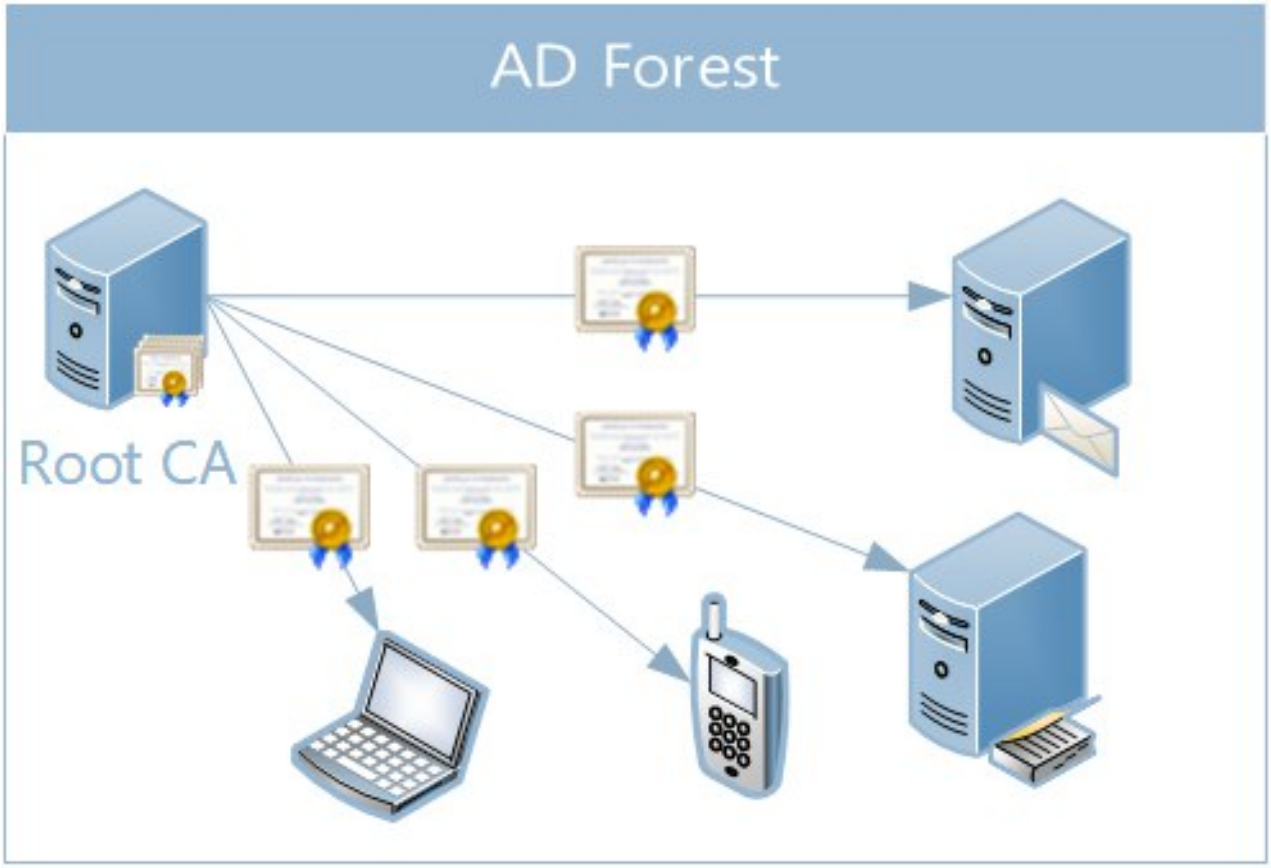
Типовые схемы развёртывания иерархии PKI

В этом разделе мы рассмотрим типовые (или классические) схемы развёртывания иерархии PKI в условиях предприятия и проводятся оценки каждой схемы и

рекомендации. Следует отметить, что ни одна из них не является универсальной, и каждая может иметь смысл в своих пределах.

Одноуровневая иерархия

Одноуровневая иерархия является самой простой в реализации и имеет следующий вид:

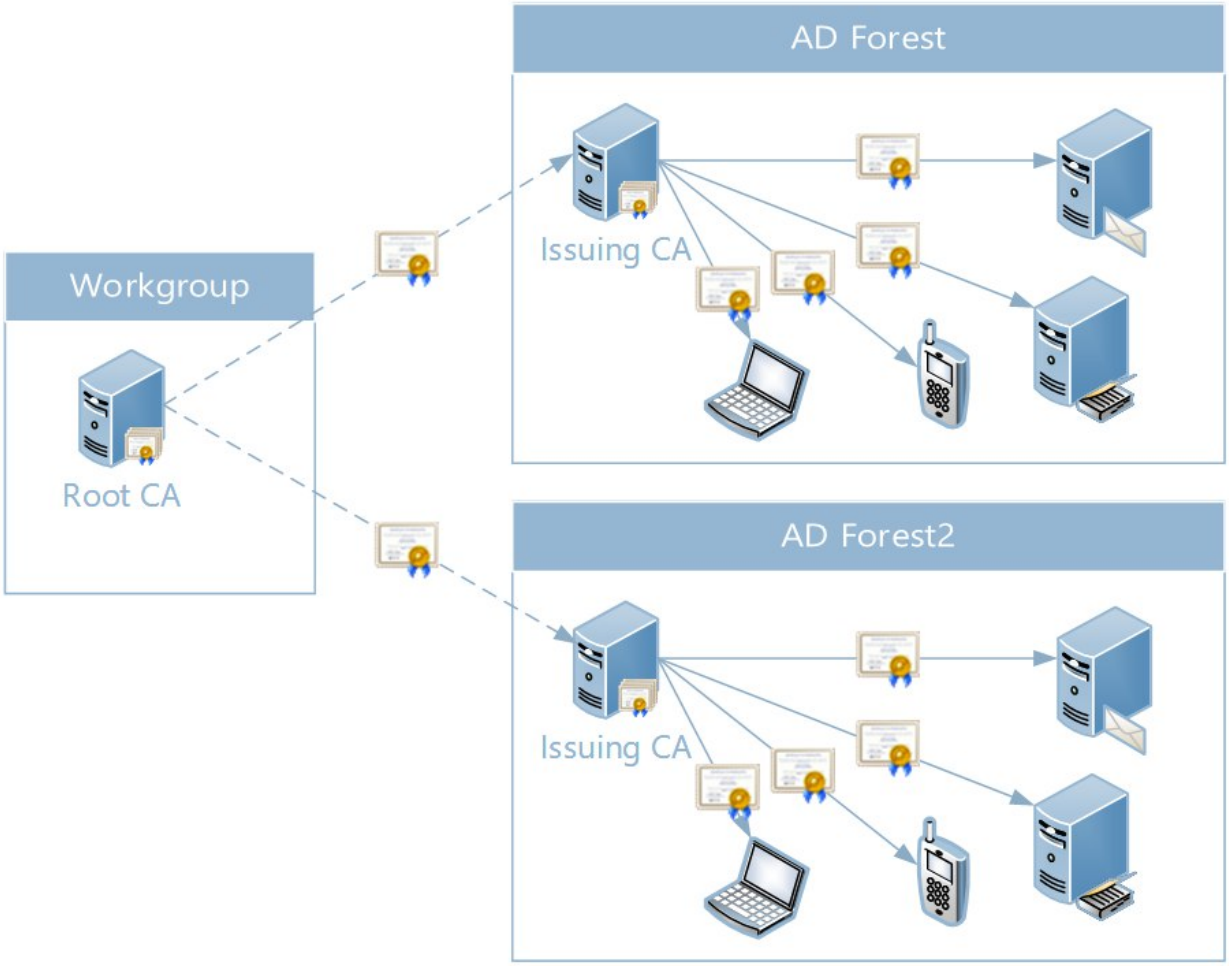


Такая иерархия является самой простой и экономичной как по ресурсам (лицензиям), так и по расходам на обслуживание и управление. Достаточно развернуть один такой ЦС в лесу Active Directory, и он будет обеспечивать сертификатами всех потребителей. Из неочевидных, но немаловажных достоинств можно отметить очень короткую цепочку сертификатов. Т.е. время на проверку доверенности и отзыва сертификата будет тратиться гораздо меньше, чем в любых других иерархиях. Однако одноуровневая иерархия обладает рядом достаточно существенных недостатков. Самый крупный из них – низкий уровень безопасности. Поскольку ЦС в такой схеме должен быть постоянно включенным в сеть, чтобы клиенты могли запрашивать сертификаты, значительно увеличивается риск его компрометации. Последствия компрометации могут быть чудовищными, вплоть до полной потери контроля над Active Directory и краху ИТ систем. Это может быть вызвано как недостаточными мерами безопасности, наличием не закрытых уязвимостей ОС или компонентах системы, которые позволяют удалённое исполнение кода и т.д. Как отмечалось выше, отозвать нормальным способом такой сертификат уже нельзя, и последствия будут действительно тяжёлыми. Другой

момент связан с гибкостью разделения ЦС на функциональные уровни (делегирование). Например, невозможно организовать несколько различных политик выдачи, разделить на классы (например, один ЦС издаёт сертификаты только машинам и устройствам, другой только для пользователей) и т.д., потому что он всего один. Недостатки одноуровневой иерархии заметно перевешивают её преимущества в лёгкости и компактности. Именно поэтому такая конфигурация имеет смысл лишь в каких-то небольших и изолированных сетях с низкими требованиями по безопасности. Например, это может быть какая-то тестовая среда. В бизнес-среде такое решение использовать не рекомендуется.

Двухуровневая иерархия

Двухуровневая иерархия уже подразумевает как минимум два ЦС в дереве, в котором один строго корневой, а остальные — подчинённые. Схема такой иерархии представлена ниже:



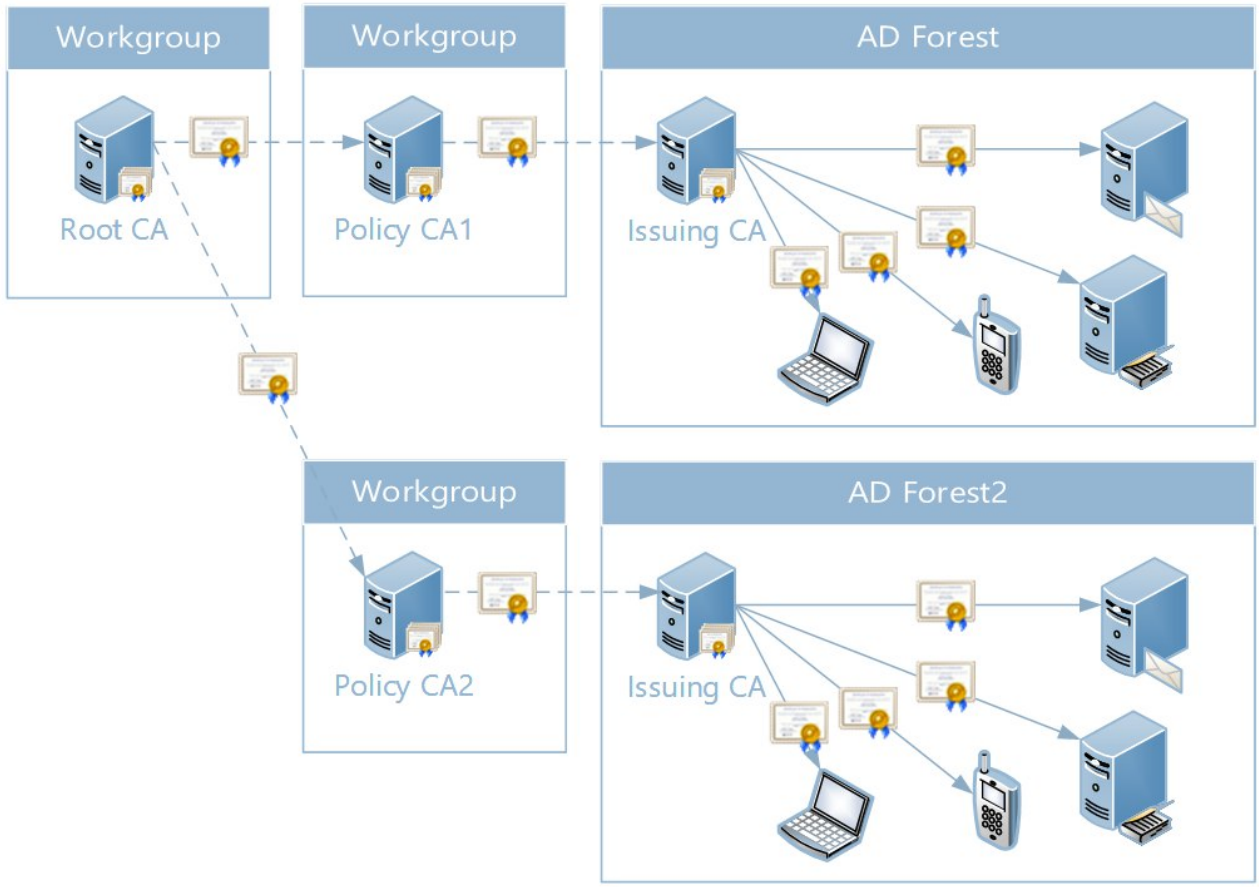
Примечание: здесь пунктирными линиями отмечен ручной (неавтоматизированный) процесс получения сертификата. Сплошными линиями отмечен автоматизированный процесс получения сертификатов. В двухуровневой иерархии уже возможно решить недостатки одноуровневой иерархии. Здесь корневой ЦС выпускает сертификаты только для подчинённых ЦС, а уже подчинённый ЦС выдаёт сертификаты конечным потребителям. Поскольку

издающие ЦС развёртываются не так часто, и срок их действия достаточного велик, это позволяет изолировать корневой ЦС от сети. Это автоматически сводит к нулю шанс компрометации такого ЦС, поскольку без сети к нему не добраться. Более того, основное время жизни он может (и должен) проводить в выключенном состоянии. Включать его нужно только для обновления собственного сертификата, подчинённого ЦС или для публикации нового CRL. В остальное время он никому не нужен. Другим достоинством двухуровневой иерархии является улучшенная гибкость в разбиении подчинённых ЦС на какие-то классы. Тот же типичный сценарий, когда два ЦС управляются разными подразделениями ИТ, и каждый ЦС выпускает сертификаты для своих групп потребителей. Например, для машин отдельно, для пользователей отдельно. Можно для корпоративных разработчиков (которые обычно живут в своих средах) выделить свой подчинённый ЦС. Именно здесь уже можно начинать задумываться о разделении ЦС по политикам выдачи (или классам). Например, можно выделить один ЦС для выдачи сертификатов с повышенными требованиями к сертификатам (например, сертификаты для аутентификации и цифровой подписи) и ЦС общего назначения. Управлять ими могут различные команды ИТ-администраторов. При этом каждый подчинённый ЦС будет совмещать задачи ЦС политики и издающего ЦС. Это вполне допустимо, если предположить, что количество ЦС на каждый класс не более одного. Из недостатков можно выделить лишь некоторое увеличение как административных, так и финансовых издержек (требуется дополнительная лицензия). К административным издержкам добавятся контроль за сроком действия каждого ЦС и списка отзыва корневого ЦС, а также их своевременное обновление. Кроме того, несколько увеличится время построения и проверки цепочек сертификатов, поскольку добавляется ещё один уровень. На практике это время практически не ощутимо. Для небольших и средних предприятий такая схема является наиболее оптимальной, поскольку она позволяет обеспечить должный уровень безопасности и приемлемый уровень гибкости разделения ЦС на определённые функции.

Трёх- и более уровневые иерархии

В более крупных компаниях со сложной организацией сетей может случиться, что двухуровневая иерархия не может обеспечить необходимый уровень гибкости управления ЦС. Например, когда компания использует географический принцип разделения с относительной автономией ИТ отделов в каждом регионе. Представьте международную компанию с региональными офисами в разных странах. В них может действовать своё законодательство в вопросах безопасности, например, при обработке персональных данных. Для соблюдения подобных юридических формальностей на каждый такой регион выделяется свой собственный ЦС политик (при этом, размещается он, как правило, в головном офисе компании). В в своём сертификате он указывает поддержку (и ссылку на соответствующий документ) тех или иных нормативных документов и выпускает сертификаты для издающих ЦС, действующих в рамках тех политик, которые обозначены в сертификате (грубо говоря, в данном регионе). В таких иерархиях

корневой ЦС и ЦС политик изолируют от сети, а к сети уже подключаются только издающие ЦС:



Здесь также пунктиром отмечен ручной процесс получения сертификата для ЦС и сплошными линиями автоматизированный процесс получения сертификата для клиентов. К плюсам такой схемы можно отнести гибкость полученной PKI с возможностью адаптации под практически любые условия. Правда, за это придётся платить. Во-первых, многоуровневые иерархии удорожают стоимость развёртывания и сопровождения PKI. Во-вторых, клиентам требуется больше времени на построение и проверки на отзыв полных цепочек сертификатов, что может вызывать отказы из-за превышения таймаутов верхних протоколов передачи данных. И на практике такие схемы зачастую являются избыточными. Поэтому при выборе подходящей иерархии ЦС следует искать баланс между гибкостью и практической целесообразностью с учётом капитальных и операционных затрат на содержание PKI. В рамках этой серии статей рассматривается наиболее популярная (в большинстве случаев) двухуровневая иерархия.

Об авторе

Вадим Поданс — специалист в области автоматизации PowerShell и Public Key Infrastructure, Microsoft MVP: Cloud and Datacenter Management с 2009 года и автор модуля PowerShell PKI. На протяжении 9 лет в своём блоге освещает различные вопросы эксплуатации и автоматизации PKI на предприятии. Статьи Вадима о PKI и PowerShell можно найти на его сайте.