Why we need multicloud IP address management and DNS orchestration

No cloud is an island

In our networked world, humans, and machines require things to be named to facilitate communication across boundaries. These names are more than labels. They are invoked to either reach or provide services, locally and remotely. Multicloud is a term often used in relation to leveraging multiple cloud providers concurrently, but it's also equally valid when talking about extending your own organization's compute boundaries into another network. Once an organization consumes services from any external cloud outside of its own network, it can now effectively be thought of as multicloud.

This means Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) extend an organization's footprint. Aspects of these remote services may have different levels of governance or underlying operational responsibility, but all resources and services must be named and reachable to provide utility. It is with this understanding that multicloud IP address management (IPAM) and DNS administration become crucial for all the digital assets you name and control.

Multicloud can then be thought of as encompassing certain extensions to your local networks and private clouds. Naturally, your first cloud, your "home" cloud, grows until IPAM in networking is required. Equally, DNS and DHCP management and orchestration needs grow as services begin to scale across new edges and clouds. DDI (DNS, DHCP, and IPAM) management then becomes crucial for almost every network flow. Visibility and unified DDI management become more than just capacity management or reporting afterthoughts; they're an operational necessity to keep the packets flowing.

The biggest SPOF (Single Point of Failure)

"It's not DNS. There's no way it's DNS. It was DNS" may make us smile, but this scenario is one we want to avoid whether it's for small-scale individual user problems or larger-scale infrastructure P1s (Priority One). The severity and impact of DNS-related incidents can disrupt everything from internal productivity to externally facing revenue-generating systems.

DNS management and a trustworthy IPAM are more than operational necessities; they're core to all phases of a project's lifecycle. Starting from early design and subsequent asset allocation, to migrations and deployments along the way, DNS and IPAM-related errors can trip up even the most pedantic, efficient, and experienced engineers. The solution is to put in place systems and processes that provide assurance but not friction, from start to finish.

AWS Route 53 and VPCs

Albeit "the cloud is someone else's computer", AWS has managed to achieve public cloud dominance with a 33% share1 of the market. Amazon is well known for the reliability, scale, performance, and flexibility of services like EC2 (Elastic Compute Cloud), S3 (Simple Storage Service), Route 53 DNS, and Virtual Private Compute (VPC), yet they are only a fraction of its overall offerings. These four fundamental building blocks are, however, some of the most frequently leveraged entry points when building with or evaluating AWS.

Whether an organization dabbles with AWS or goes all-in means that throughout any engineer or manager's career, they inevitably will face critical dependencies on Route 53 for DNS services and VPCs for network containers. Although AWS seems inescapable, it's not the only cloud in town, but it is the go-to for most when extending, migrating, or growing compute assets into a hybrid or multilcloud architecture.

What drives the use of multicloud DNS

Expansion, diversification, and new technologies

As any company or organization grows, so too does its IT footprint and associated technical stack. This growth is rarely uniform. It can often look lumpy as a result of competing priorities, scarce resources, and fluctuating constraints. For organizations that aren't growing, they do not wish to stagnate either and often use windows of opportunity to address technical debt or improve existing workflows and processes. Irrespective of either scenario, IT teams are regularly expected to have made decisions and deployed modular designs that facilitate both organic and projected growth. As each organization expands and morphs over time, technology itself also changes and improves. This results in finite lifetimes for compute, network, and software assets.

Additionally, as functionality requirements change across services, groups, and locations, IT footprints naturally fragment, balkanize, and then finally rejoin. This leads to multiple phases and technology trajectories existing concurrently within organizations. For medium to large organizations, this is to be expected. It is in some ways an outgrowth of Conway's Law as organizations develop. Practically speaking though, it leads to a need for flexible OTT (Over The Top) solutions that preserve visibility and provide unified management of core atomic assets. Just as finance departments are increasingly expected to communicate real-time top and bottom lines, so too must IT be able to audit, allocate, and account for their DDI (DNS, DHCP, and IPAM)-related virtual assets.

Independent of whether waterfall or agile methodologies are used, IaC (Infrastructure as Code) and IaaS (Infrastructure as a Service) are shining a spotlight on the importance of timely and accurate virtual asset management, especially within DDI. From the data center to the access edge, and laterally across all business units and geographies, heterogeneous environ-ments are accelerating, all the while becoming more ephemeral and malleable. When combined with human fallibility and mere mortal response times, manual processes must be replaced with resilient, intelligent, and automated workflows. Change must indeed still be managed, but faster and with ever-greater assurance so human gatekeepers can quickly decide or delegate.

Mergers and acquisitions (M&A)

Another crucial consideration for multicloud DNS and IPAM relates to M&A (mergers and acquisitions). If during an acquisition, new footprints of assets and records must be onboarded for visibility, management, and orchestration, the acquiree's IP and DNS spaces must be integrated somehow. This integration may not entail renaming assets but often includes duplicate IP prefixes from the acquiree's networks and public cloud infrastructure. Even the most flexible designs and namespaces can sometimes fall foul of acquisitions that demand tracking multiple duplicate IP spaces spread across disparate and heterogeneous clouds. Acquisitions, be they large or small, can result in a rapid onset of multicloud data management, many of which entail multiple new AWS accounts.

AWS as an Inevitable Outgrowth

Amidst the behemoth that is AWS and its growing influence over cloud compute, we must remember that most organizations' physical and digital footprints still exist in many locations and are built using a range of vendors. These diverse physical and virtual assets may inhabit an organization's networks anywhere along the path from access edge to WAN (Wide Area Network) or even inside fabrics that form on-premises private clouds. Public clouds are often then seen as greenfields and blue skies for new projects and cost-saving initiatives. Ultimately, and without the right unified approach, public cloud usage can rapidly and more quickly become the new brownfields.

Whereas some startups and cloud-native organizations have gone all-in with common AWS offerings, it doesn't preclude them from having traditional digital and physical assets in offices, campuses, and other commercial premises. The "cloud" must be accessed from somewhere and by something for it to have utility. Machines, agents, and humans all depend upon DNS, DHCP, and IPAM (DDI) to get from one cloud to the other.

Key considerations for DDI automation, orchestration, and management

In a world where IP addresses, prefixes, and virtual assets define our digital borders and flows, it can be a mammoth undertaking to track, allocate, and assure ourselves our fundamental building blocks are sound. This challenge is exacerbated when we introduce IaC (Infrastructure as Code) and programmatic access that requires DNS and IPAM services. Automated service consumption, whether by external customers or internal users, means everything moves faster and heightens the requirement for DNS management and AWS IPAM records to be fresh, correct, and accurate.

Automation and API-driven cloud services accelerate the game, as does SASE (Secure Access Service Edge) leveraging SD-WAN (Software-Defined WAN). We begin to more deeply realize that DNS is one of the most, if not the most important, active asset. It facilitates resource location, defines service endpoints, uncovers paths to dependencies, and enables both public and private service consumption. A unified DDI solution allows for one trusted System of Record (SoR) to exist. It talks to multiple Sources of Truth (SoT) while providing a single authoritative UI (User Interface) and API. Integrating, allocating, building, and querying is then simplified and streamlined, no matter the silo.

With DNS at the heart of any distributed digital footprint, the cadence of change means operations and security teams must be furnished with the right tools and data to maintain quality and do meaningful work. Service Level Objectives (SLOs) can only be met when the level of visibility and observability facilitates virtuous cycles and thus continuous improvement. Then, whether you deploy on Fridays or not becomes moot once the quality of changes are assured. Automation and change velocity is predicated on good record management. It then accelerates the adaptability of IT and change in the organization itself.

Supported platforms and endpoints

When choosing a DDI solution, it's imperative that it can meet more than your current needs. It must also be able to evolve with changing needs. One way to ensure this is by selecting a DDI platform that doesn't try to replace existing best-in-breed DNS or DHCP service solutions. Friction must be avoided at all costs, so the solution should allow for OTT (Over The Top) management and orchestration of existing services, thus preserving existing investments and expertise while providing additional value with a new unified and trusted authoritative layer.

This OTT architecture can only be achieved once a wide range of existing DNS and DHCP platforms are fully supported. Providing support for commonly used services such as Windows DNS, BIND, Unbound, PowerDNS, Akamai Fast DNS, Azure DNS, Amazon Route 53, NS1, and Dyn, to name a few, means avoiding rip-and-replace projects and enjoying accelerated time-to-value. For the aforementioned cloud providers, multicloud data management becomes pivotal for development, staging, or production environments as they extend from your home cloud out to other cloud providers

Workflows and features

In the context of getting things done, work often takes place following formal or, more often than not, informal workflows. Formal workflows, by definition, follow a model with well-known states. Informal workflows are effectively carried out in the dark and can bite organizations hard when dealing with DNS, DHCP, and IPAM tasks. Informal workflows often require mapping and modeling before any optimizations or automation can begin. Both can benefit from increased auditability and the insertion of frictionless Change Management.

When supporting a wide range of DNS and DHCP services, it's crucial that any management and orchestration platform provides its own add-value feature set. When embracing an OTT (Over The Top) DDI solution, best-of-breed services can evolve independently and at different rates at the service edge, while the core platform and data types supported can evolve more steadily and in a more agnostic manner. This means that an OTT DDI solution can focus more on add-value macro features and workflows (rather than trying to re-invent the wheel and then having to debug it at the service edge). The OTT model also facilitates incredibly useful features like xDNS to provide redundancy, smooth migrations, and defensible partitions with zones spread across multiple service

providers. When tying features and workflows together, the added capability of extending and programming an environment becomes a true force multiplier. With a first-class API, not only can workflows reach beyond one system or silo, they can be integrated into a whole range of productivity, notification, and ticketing systems. Security and operations teams can also automate incident response and enrichment procedures that rely upon a trusted SoR (System of Record) covering all an organization's DDI assets.

AWS and The Micetro REST API

In addition to Micetro's powerful UI, its fully-featured API also facilitates AWS Route53 and VPC actions. The Micetro API acts as a single broker to unlock and enable greater extensibility and automation. By providing a single-layer API that abstracts DDI tasks and actions across multiple providers, teams are empowered to enhance their own workflows, build integrations, and create custom solutions.

The downstream AWS complexity and required glue are fully abstracted away. Common tasks are then initiated from Micetro as a unified and authoritative source. This enables local and remote teams to be more efficient, make smarter decisions, and collaborate across project and team boundaries, all of which lead to increased innovation.

Once you install the free trial, there is a full set of OpenAPI (Swagger) documentation available directly on your Men&Mice web application accessed via the path **/mmws/api/doc/**, and the latest product guides are always available at https://menandmice.com/docs.

Multicloud data management and integration with AWS

How to Integrate Micetro with AWS Route 53 and AWS VPCs

Micetro is an overlay and orchestration solution for DDI (DNS, DHCP, and IPAM) environments, including on-premises and cloud-based assets. Two of AWS' pivotal services are Route 53 for DNS and VPC (Virtual Private Compute) for defining virtual IP networks. This will let you take advantage of a unified and consolidated System of Record (SoR) that encompasses all your DNS footprints (and associated Sources of Truth (SoT)).

Micetro also provides workflows, reporting, and a fully-featured API layer. Micetro simplifies all DNS, DHCP, and IPAM operations with a platform that integrates and unifies heterogeneous environments without replacing them. It uses an OTT (Over The Top) architecture to minimize upheaval and maximize visibility.

First, we will focus on ensuring the correct elements are in place for a Micetro installation to be able to talk to AWS (including any configuration items required on either end) to facilitate:

- administration of zones and zone records
- administration of VPCs (and their characteristics)
- (automated) asset and record synchronization
- workflows (with built-in Change Management)

Note: We will be using a subdomain called "aws" in our demo domain, "menandmice.cloud". This will allow us to clearly partition our assets and records for the reader, as other guides will focus on Microsoft Azure, Akamai Fast DNS, Openstack, etc. Your organization's specific namespace and DNS architecture will look different, of course. Micetro can integrate with and administer a whole range of scenarios and architectures. If you do require any help, have any questions, or would like some training, please reach out to us here.

Download: The following technical sections will be using version 10.2.2 of the Micetro suite. A fully functioning version is available as a free trial and does not require payment details.

Ensuring Micetro is ready for Cloud Services

Micetro is comprised of a suite of software services. The most important element of that suite is called Men&Mice Central. Men&Mice Central is the heart and soul of Micetro. All the other elements leverage it in one way or

another. We need to ensure we have the following four elements to facilitate integration with AWS (which requires Micetro's Cloud Services component):

1. Men&Mice Central is required first and foremost (as mentioned). It runs on Windows or Linux and is the main hub of Micetro. It performs roles such as Identity and Access Management, database, and API access, among others...
2. Men&Mice Web Application is the primary UI (User Interface). It is common practice to run it on the same host as Central (but this is not required) though it does require a web server like Apache or IIS to be running on the same host. It also runs on Windows or Linux.
3. Men&Mice Console is required for the initial Micetro setup (after software installation), some system administration, and certain features. All functionality is currently being integrated into the Web Application. It only runs on Windows.
4. Men&Mice Server Controller is required to broker the connection to AWS. These DNS Server Controllers co-locate with BIND, Active Directory, or other DNS servers such as PowerDNS on the relevant underlying operating system.

Note: We will not be walking through the basic installation of each element (1-4) above, but rather how to connect the Cloud Services component to AWS and administer assets using it. If you need any installation help, there are detailed guides available here. We will link to more detailed documentation where applicable, but https://menandmice.com/docs always points to the latest version. Please pay attention to the required UDP and TCP port communication between elements.

Note_II: We also need to ensure that the correct licenses are installed for the DNS and IPAM modules. Additional licenses (all available from the free trial) are required for the advanced Workflow and Reporting modules.

AWS access prerequisites and current limitations

Once integrated, Micetro will import all zones and records from AWS Route 53 (and also be able to create and edit them). If additionally configured for AWS VPC integration, Micetro will also learn about subnets assigned to VPCs, their usage, and their characteristics. It will also populate the Micetro IPAM with this data. To achieve this, Micetro requires API access to Amazon AWS using an API Access key. There are detailed instructions for all cloud integrations here, but a brief AWS-specific summary follows below.

Note: The primary Micetro Central host running a Micetro DNS Server Controller must be able to speak outbound to Amazon AWS using the port TCP 443.

AWS API key, policies, and permissions

An AWS user account is required for Micetro to engage with and administer assets and records. This user account's credential type is that of a programmatic access key, and it will be required in the Micetro Console to configure cloud services access. This key is used with the AWS API and requires the correct policies and associated permissions to administer AWS Route 53 for DNS and/or Amazon EC2 for VPCs.

For a single Amazon AWS account integration, the following IAM policy names are required for the access key:

- "AmazonRoute53FullAccess" for DNS
- "AmazonEC2FullAccess" for VPCs
- "IAMReadOnlyAccess" for related aliases/lookups

For a multi-account Amazon AWS integration, the same policies are required, but the user account must be part of a group capable of adopting an STS role from other accounts. These roles must have the correct permissions also:

- "AmazonRoute53FullAccess" for DNS
- "AmazonEC2FullAccess" for VPCs
- "IAMReadOnlyAccess" for related aliases/lookups.

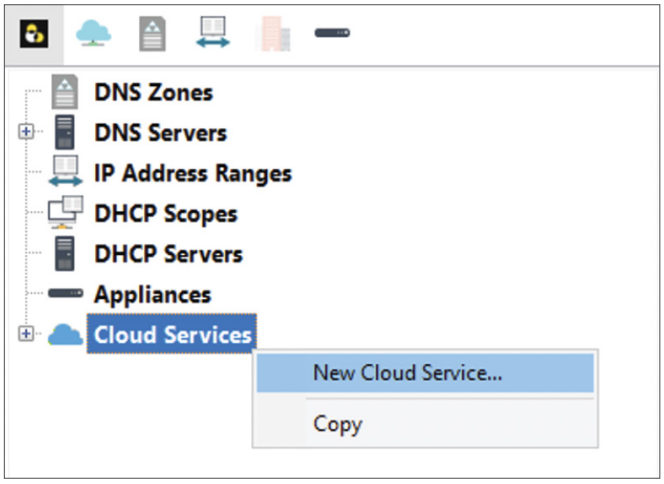
In the multi-account setup, the user account must be added to a new group with its own inline custom group policies. Each policy then allows for the Action "sts:AssumeRole" for a Resource that contains the ARN of and named role for the additional accounts. This means that each custom group policy will have a slightly different Resource ARN based upon the ID of the alternate account and how you've chosen to name the role in that account. More information on configuring multi-account AWS access can be found here.

Note: Amazon VPCs are currently only administrable via the Men&Mice Console or API. This will be addressed shortly as the Men&Mice web application achieves full feature parity with the Men&Mice Console. There may also be some warnings in the Men&Mice Console about subnet collisions as the same RFC1918 IP prefixes can be used in different VPCs across multiple availability zones and AWS regions.

How to add AWS as a cloud service in the console

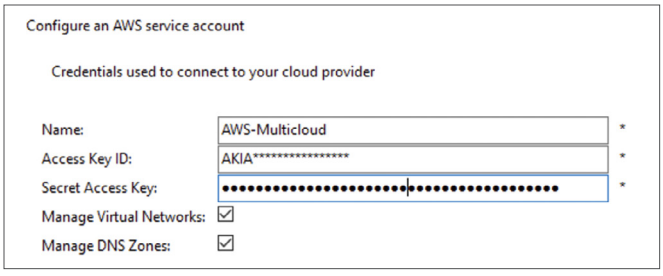
Once you have the Men&Mice Console installed and talking to Men&Mice Central, you can use the AWS API key from the previous section to add AWS as a "Cloud Service".

In the Men&Mice Console, go to "Cloud Services", and with a right-click, select "New Cloud Service".



Here we can then select the type of cloud service required:

We then want to Name our connection, enter a specific Access Key ID, its specific Secret Access Key, and whether to Manage Virtual Networks (essentially VPCs) and/or Manage DNS Zones. The configured key must have the correct policy names and permissions, as previously mentioned.



Micetro will retrieve the data from the cloud provider, save the account information, and configure its services. Micetro will then synchronize with AWS every 900 seconds, and we are ready for a range of common tasks, workflows, and single-layer API-driven automation across our DNS namespace and IP footprint.

Note: All sensitive data required for communication with AWS, such as API keys and secrets, are encrypted both at rest and in transit.

Common tasks for an AWS cloud DNS integration

Simplifying and securing operations

With a unified OTT (Over The Top) orchestration and management platform such as Micetro, all DDI (DNS, DHCP, IPAM) administration tasks, related access, and visibility can be centralized and controlled. Roles and responsibilities can be allocated across departments, teams, individuals, or technologies using groups and roles. Implementing a "least access principle" is easy, and it protects assets and asset classes from unauthorized changes that may lead to resource exhaustion or unexpected outages.

With integration across your full heterogeneous DDI footprint, it's now possible to get unprecedented visibility, auditability, and control from a single unified platform. Although management and orchestration are performed by Micetro, each individual entity, such as Route53, BIND, or Microsoft AD, continues to provide its own services. This ensures robustness and resilience across services your team already knows well and relies on, which makes everything simpler to manage, results in fewer errors, and ensures better outcomes.

Fine-grained access management with RBAC

In Micetro, Role-Based Access Control (RBAC) is applicable to more than just macro-level services like DNS, DHCP, IPAM, and reporting. RBAC is also applicable to specific and individual assets managed by Micetro. This allows for additional and exceptional fine-grained controls at an individual network, container, or zone level (if so desired).

Whether you want to use the default out-of-the-box groups and roles, build on them, or start from scratch to create custom roles, you can be coarse with some groups or pedantic with others. There's also a range of primitives including but not limited to; create, add, read, list, edit, delete, use, enable, and release that can be applied to objects (and their sub-objects) to create roles governing:

- DNS servers
- DHCP server
- DNS zones
- Ranges and DHCP scopes
- Address spaces
- Cloud networks
- Cloud services

Additionally, roles for Micetro itself relating to general administration and additional access to elements such as Reporting and the Workflow module can be configured and applied with many default groups available out of the box.

Optional workflows for change management

One of the huge benefits of an OTT and unified DDI platform is that the whole DNS namespace and related IP addresses can be managed centrally. Micetro provides an optional Workflow module that streamlines DNS Change Management. It can be used for the creation, modification, and deletion of Resource Records (RRs).

This feature means that any Micetro user (or group) given the Requester (built-in) role, such as project teams, helpdesk technicians, or other engineers, can raise DNS change requests. The requests then require approval from an administrator with the appropriate Approver (built-in) role. Once approved, changes are automatically implemented. Open requests can be easily viewed and amended before being submitted for approval. Requests

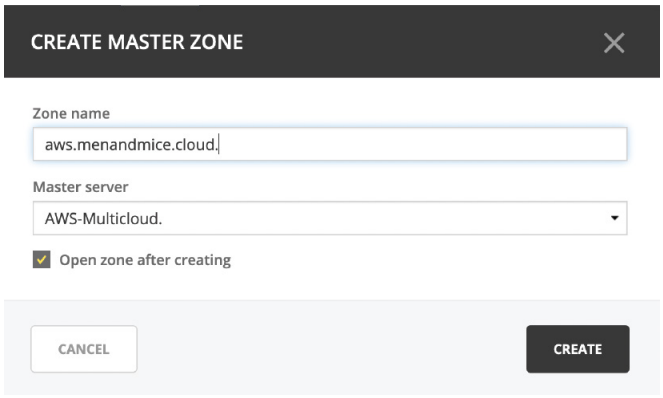
move through basic states such as Open, Pending, Failed, or Closed (where Closed may mean Scheduled, Fulfilled, or Rejected).

Changes can even be scheduled during the request creation stage. Once approved, they are then automatically applied at the requested time and date. So, anyone with the Micetro Requester access can raise changes, get them approved, and have them made live in AWS Route 53, all without having to know anything about Route 53 and without needing any AWS console or AWS API access. Micetro's Workflow module is equally applicable to any of the other supported DNS servers and cloud providers.

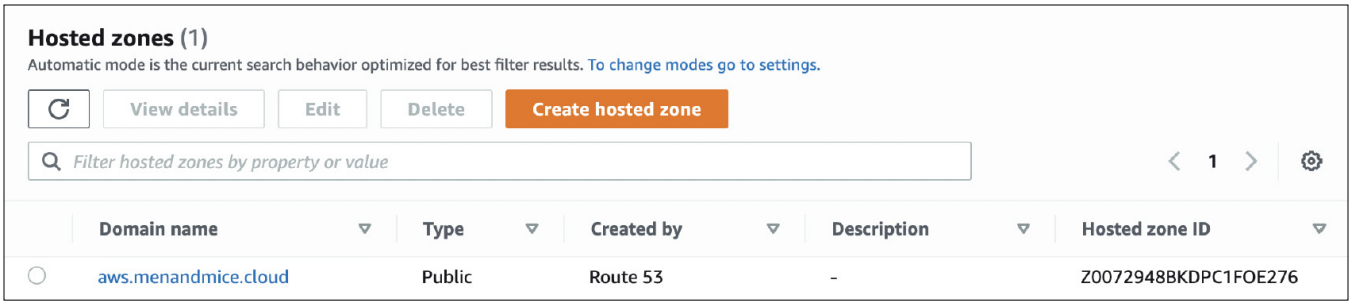
How do I change DNS in AWS?

Creating zones in AWS Route 53

From the Micetro web application, it is trivial to create master zones in Route 53. Once you have DNS administrator privileges, you just go to the "DNS" section and then "Create/master zone". Enter the full Zone name (not forgetting the trailing dot), e.g., "aws.menandmice.cloud." and ensure you select the correct Master server, which, in our case, is the cloud service called "AWS-Multicloud".



To validate this, once submitted, you can see that the zone has been created in AWS Route 53 (below) including the SOA and NS records.

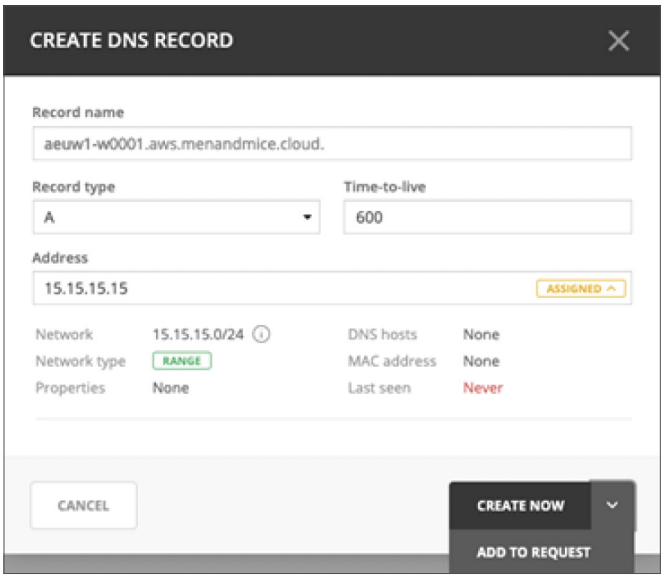


Tip: If you delete and recreate a zone in AWS, always remember to check your NS records are correct if you are delegating from outside of Route 53.

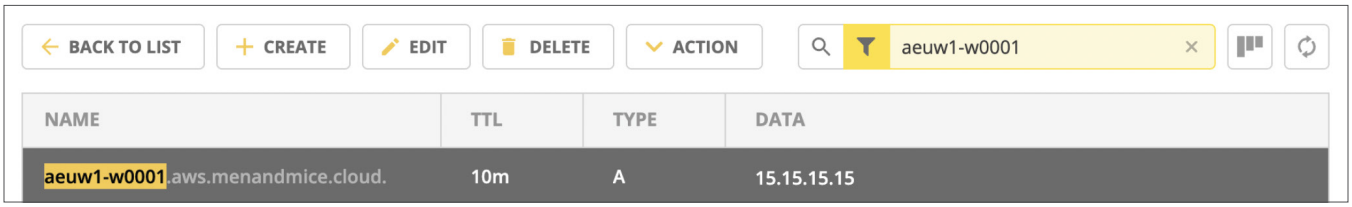
In Micetro, you can subsequently perform actions on the zone such as "Open", "Migrate", and "Delete" while also assigning fine-grained access permissions via Micetro's RBAC by selecting "Access". You can also see the full administrative history of the zone.

Creating / editing resource records in AWS Route 53

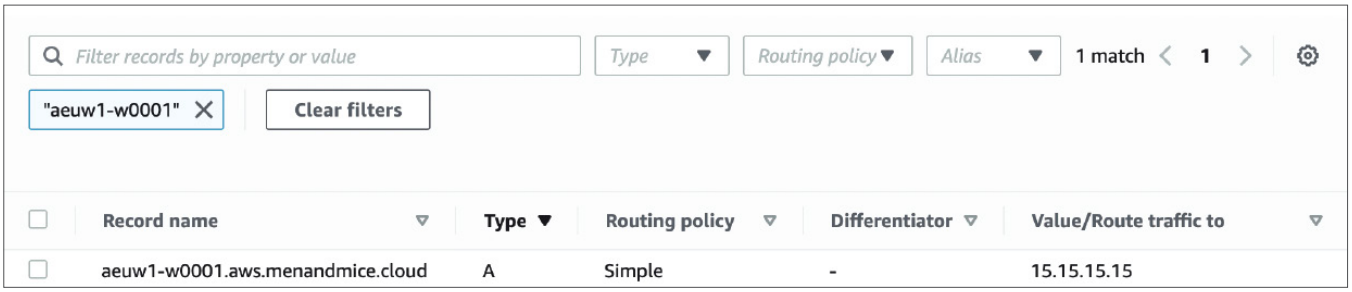
Now, let's create some records in our newly delegated zone of "aws.menandmice.cloud.". As a DNS administrator, we can choose to create and action a request immediately, or if we're a Requester (built-in), we can submit this as a change request.



Once it has been approved and actioned (if you're a DNS Administrator, you can action immediately), we can see that the record has been created in Micetro:



And in AWS Route 53:



So, from a single UI (or API) we can administer records across all our cloud services and DNS servers.

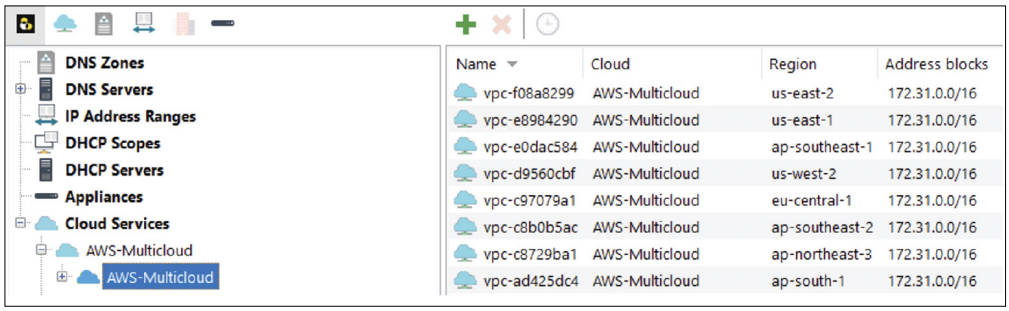
How do I set up a VPC in AWS?

Creating VPCs

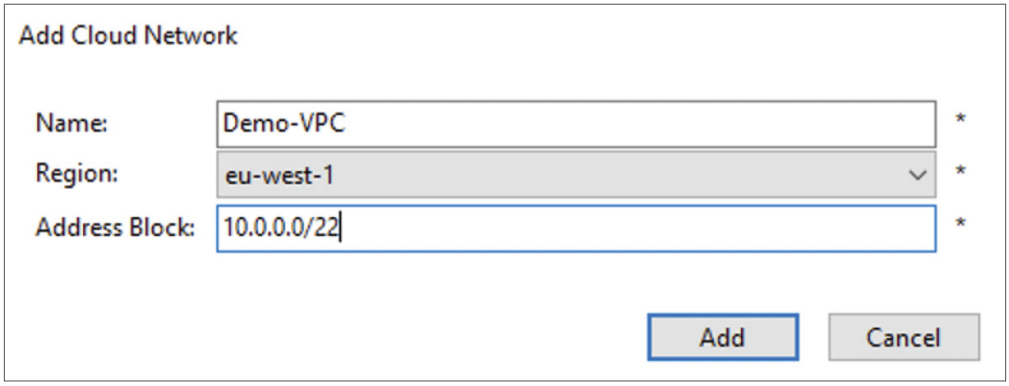
Micetro makes this easy, and it will also track our IP allocations via the IPAM. Let's create a VPC in our "eu-west-1" region using the CIDR block of "10.0.0.0/22" from which we will allocate two separate subnets of "10.0.0.0/24" and "10.0.1.0/24" from the lower /23.

Note: Currently, we use the Men&Mice Console to create VPCs in AWS, but shortly this will become available in the web application, including being applicable for the Workflow module.

From the Console, select the appropriate cloud service and then use the green plus (or Ctrl+n) to add a new Cloud Network:

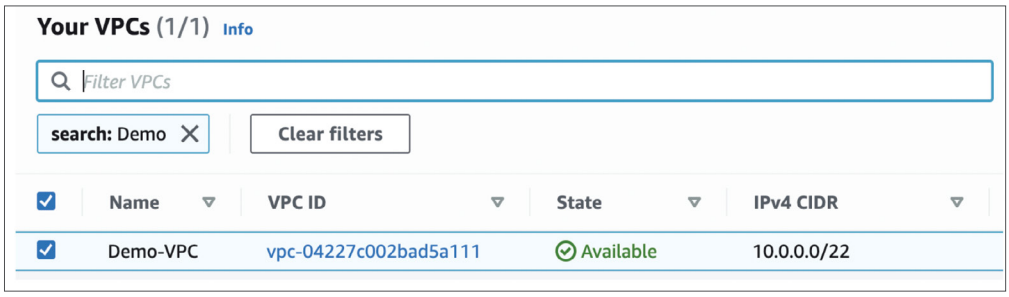


Enter the Name, Region, and Address Block required and click Add:

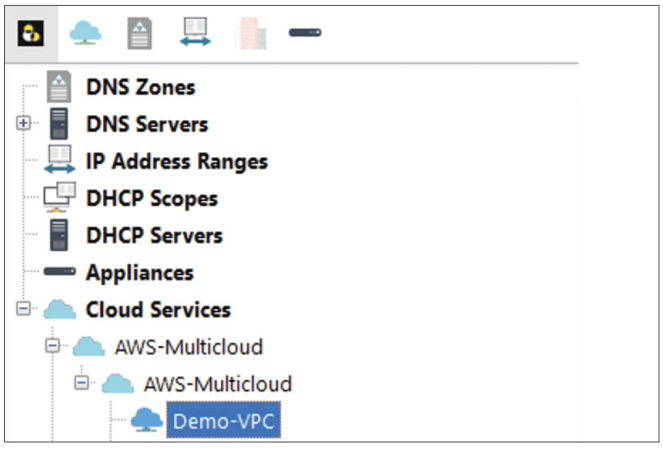


And we can now see our new Demo-VPC listed in Micetro:

But we can also see Demo-VPC is now available and ready for action in AWS:



We will now create two subnets inside our newly formed Demo-VPC. We use the green plus button (or Ctrl+n) again to add the details we want for each of the new subnets using the details below:



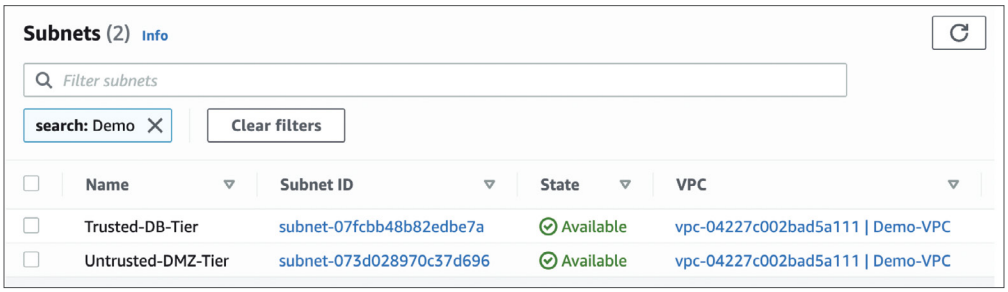
Details (subnet 1):

- Subnet: 10.0.0.0/24
- Title: Untrusted-DMZ-Tier
- Description: This is our example initial DMZ tier within the VPC.
- Cloud Network: Demo-VPC
- Address Range: Reserve Network and Broadcast Address

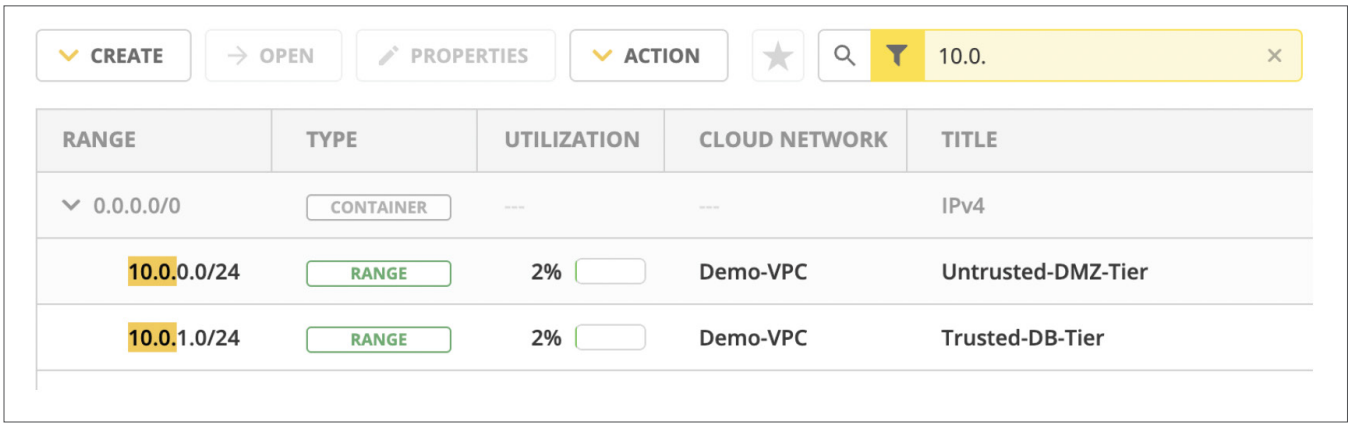
Details (subnet 2):

- Subnet: 10.0.1.0/24
- Title: Trusted-DB-Tier
- Description: This is our example trusted DB tier within the VPC.
- Cloud Network: Demo-VPC
- Address Range: Reserve Network and Broadcast Address

This results in our subnets also going live in AWS VPC:



Meanwhile, we can see that our subnets are also being tracked in our unified Micetro IPAM:



How can I use the Micetro API to change DNS in AWS?

Micetro REST API authentication

The Micetro REST API uses basic HTTP user authentication based upon the user account privileges configured. Privileges are set via the Micetro RBAC in Access Management. It is recommended to create a separate user account for use with the API.

Tip: The API user account should be scoped to only have the rights to perform the tasks required. It should not be re-used or shared for access from multiple platforms.

Example API call(s)

You can test the Micetro REST API using cURL with the appropriate user credentials to administer and access objects.

Tip: When using methods other than the default HTTP GET, remember to supply the header Content-Type of "applica-tion/json". We will use cURL to demo some API calls. cURL is available by default in macOS and most Linux distributions, but it is also available as a binary for Windows if you're using Windows and don't want to use Powershell.

Let's use the Micetro API to look up the earlier IPAM record (15.15.15.15) we created (when we added an A record to the "aws.menandmice.cloud." zone):

Using macOS/Linux:

```
curl -silent --user <user>:<password> -X GET \
"https://<your_host>/mmws/api/IPAMrecords/15.15.15.15"
```

Using Windows Powershell:

```
$cred = Get-Credential
Invoke-RestMethod -Method GET -Cred $cred -Uri
"https://<your_host>/mmws/api/IPAMrecords/15.15.15.15" | ConvertTo-Json
-Depth 5
```

Which results in the following JSON blob response containing all the associated data for the IP record (including its DNS records and DHCP information if applicable):


```
{
  "result": {
    "ipamRecord": {
      "addrRef": "IPAMRecords/30",
      "address": "15.15.15.15",
      "claimed": false,
      "dnsHosts": [
        {
          "dnsRecord": {
            "ref": "DNSRecords/42",
            "name": "aeuw1-w0001.aws.menandmice.cloud.",
            "type": "A",
            "ttl": "600",
            "data": "15.15.15.15",
            "comment": "",
            "enabled": true,
            "dnsZoneRef": "DNSZones/10",
            "customProperties": {}
          },
          "ptrStatus": "Unknown",
          "relatedRecords": []
        }
      ],
      "dhcpReservations": [],
      "dhcpLeases": [],
```

```
      "discoveryType": "None",
      "lastSeenDate": "",
      "lastDiscoveryDate": "",
      "lastKnownClientIdentifier": "",
      "device": "",
      "interface": "",
      "ptrStatus": "Unknown",
      "extraneousPTR": false,
      "customProperties": {},
      "state": "Assigned",
      "usage": 4
    }
  }
}
```


Now, let's look for the related A record in DNS by specifying the zone, but by only using a partial search term, to see what we can find:

Using MacOS/Linux:

```
curl -silent --user <user>:<password> -X GET \
"https://<your_host>/mmws/api/DNSZones/aws.menandmice.cloud./DNSRecords?
filter=type=A AND name=@aeu"
```

Or Windows Powershell:

```
$cred = Get-Credential
Invoke-RestMethod -Method GET -Cred $cred -Uri
"https://<your_host>/mmws/api/DNSZones/aws.menandmice.cloud./DNSRecords?
filter=type=A AND name=@aeu" | ConvertTo-Json -Depth 5
```

Which results in (1) record returned:


```
{
  "result": {
    "dnsRecords": [
      {
        "ref": "DNSRecords/42",
        "name": "aeuw1-w0001",
        "type": "A",
        "ttl": "600",
        "data": "15.15.15.15",
        "comment": "",
        "enabled": true,
        "dnsZoneRef": "DNSZones/10",
        "customProperties": {}
      }
    ],
    "totalResults": 1
  }
}
```

Now, let's create a totally new DNS A record in our zone "aws.menandmice.cloud." using the name "aeuw1-w0002" and the IP address "15.15.15.16", and then let's check it has propagated to AWS Route 53 and beyond!

First we shall create a small JSON blob in a file called "new.json":

```
{
  "dnsRecords": [
    {
      "name": "aeuw1-w0002",
      "type": "A",
      "ttl": "600",
      "data": "15.15.15.16",
      "enabled": false,
      "aging": 0,
      "dnsZoneRef": "DNSZones/10"
    }
  ],
  "saveComment": "Created by API"
}
```

Now let's send this data from "new.json" to our endpoint "DNSRecords":

Using MacOS/Linux:

```
curl --header "Content-Type: application/json" \
  --request POST \
  --user <user>:<password> \
  -d @new.json \
  https://<your_host>/mmws/api/DNSRecords
```

Or Windows Powershell:

```
$cred = Get-Credential
$body = @(
    @{
        filename = 'newer.json'
        filecontent = [io.file]::ReadAllText("new.json")
    }
)
$body | ConvertTo-Json
Invoke-RestMethod -Method POST -Cred $cred -Body $body.filecontent
-ContentType 'application/json' -Uri
"https://<your_host>/mmws/api/DNSRecords"| ConvertTo-Json -Depth 5
```

Which should return a HTTP "201 Created" status and the Object Reference (objRef) with no listed errors. We are now pretty happy that a record was indeed created.

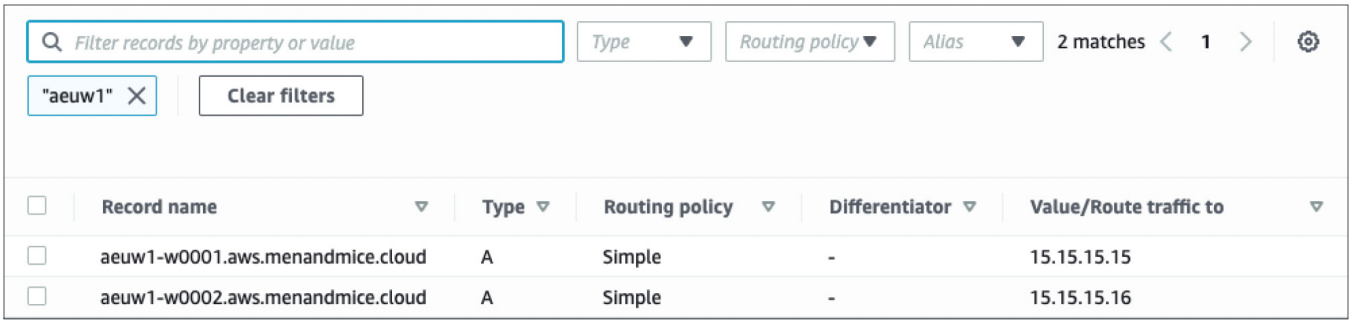
```
dig @8.8.8.8 +short aeuw1-w0002.aws.menandmice.cloud
15.15.15.16
```

Now, let's check in Micetro, AWS Route 53, and from the Internet's perspective:

Micetro looks good:

NAME	TTL	TYPE	DATA
aeuw1-w0001.aws.menandmice.cloud.	10m	A	15.15.15.15
aeuw1-w0002.aws.menandmice.cloud.	10m	A	15.15.15.16

AWS Route53 looks good:



The Internet says yes!

```
dig @8.8.8.8 +short aeuw1-w0002.aws.menandmice.cloud
15.15.15.16
```

Available Endpoints and Documentation

Once you install the free trial, there is a full set of OpenAPI (Swagger) documentation available directly on your Men&Mice web application accessed via the path **/mmws/api/doc/**, and the latest product guides are always available at https://menandmice.com/docs.

Corporate Headquarters

4100 Yonge St. 3rd Floor, Toronto, ON, M2P 2B5
1-416-646-8400 | 1-866-895-6931

bluecat.com

Next steps

Use an overlay and orchestration solution to regain network visibility and control.

Request a free trial