GSoC 2019 Proposal
Organization: CNCF (coredns)

Student Info:

- **Name:** Palash Nigam
- **GitHub username:** @palash25
- **Email:** npalash25@gmail.com
- **Location:** Lucknow, U.P. , India.
- **Time Zone:** UTC+05:30
- **GSoC blog RSS feed URL:** @npalash25 (Medium account)

Contributions to coredns:

Issue	Description	PR	Status
#1407	plugin/secondary: add metrics	#2550	Under review
-	fix link to whoami plugin page	#139	Merged
#1520	plugin/rewrite: Add metrics	#2767	Under review
#2695	Dedup code between grpc and forward plugin	#2771	WIP

Project Info:

Title	Azure DNS backend for coredns

Mentors	Yong Tang @yongtang

Abstract:

CoreDNS is able to serve DNS with cloud vendors (such as AWS) as the backend. The feature is very much useful in a hybrid environment where cloud-vendor specific service endpoints need to be exposed to clusters managed by Kubernetes. The goal of this project is to support Azure DNS (similar to already supported route53 plugin) as a backend for CoreDNS.

Project Deliverables and Goals:

This project aims to achieve the development of a fully functional, unit-tested plugin for azure dns backend that is able to serve zones from resource record sets stored in azure dns. Support for various cloud providers will enable the CoreDNS project to gain more adopters and users and provide developers with more options of providers to run CoreDNS on.

Implementation Details:

The plugin will support the following record types that are also supported by the azure golang sdk

- A
- AAAA
- CAA
- CNAME
- MX
- NS
- PTR
- SOA
- SRV
- TXT

To make sense of this project I have been going back and forth between the current implementation of the route53 plugin and the azure dns api docs to find similar methods used for listing zones and resource record sets in the azure API.

The work can be divided into three main chunks:

1. Reading the azure credentials and the plugin config from the corefile.
2. Fetching RecordSets and Zones using the Azure API
3. Writing tests and docs for all the code implemented

Reading the config from the Corefile is pretty standard and easy to implement as it more or less the same in most of the plugins. The focus of this phase would be to write a mechanism to read the Azure credentials in a ***ChainCredential*** fashion similar to the route53 plugin i.e.
Static Azure keys -> Environment Variables -> Credentials file -> IAM roles

The azure API makes this possible by providing us 4 ways of authentication using credentials out of which 3 are going to be used here

1. Client Based Authentication: where we will read the creds from the azure block in the Corefile and authenticate a client with it our plugin code

```
import "github.com/Azure/go-autorest/autorest/azure/auth"
certificateAuthorizer := auth.NewClientCertificateConfig(certificatePath,
certificatePassword, clientID, tenantID)
authorizerToken, err := certificateAuthorizer.Authorize()
```

2. Environment Based Authentication: reads the credentials from env variables

```
import "github.com/Azure/go-autorest/autorest/azure/auth"
authorizer, err := auth.NewAuthorizerFromEnvironment()
```

3. File Based Authentication: reads from a credentials file.

```
import "github.com/Azure/go-autorest/autorest/azure/auth"
authorizer, err :=
NewAuthorizerFromFile(azure.PublicCloud.ResourceManagerEndpoint)
```

Once this is complete the setup part of the plugin will be done.
The second phase deals with the fetching of resource record sets and the DNS zones from Azure so that they can be served from CoreDNS. Looking through both the aws and azure go sdk some parallels for such methods were discovered.
Azure provides

```
func (client ZonesClient) List(ctx context.Context, top *int32) (result ZoneListResultPage, err error)
```

Which lists the DNS zones in all resource groups in a subscription

Here is a sample request and response

Request:

```
GET
https://management.azure.com/subscriptions/subid/resourceGroups/rg1/providers/Microsoft.Network/dnsZones?api-version=2018-03-01-preview
```

Response:

```
{
 "nextLink":
"https://servicehost/subscriptions/subid/resourceGroups/rg1/providers/Microsoft.Network/dnsZones?api-version=2018-03-01-preview&$skipToken=skipToken",
 "value": [
   {
     "id":
"/subscriptions/subid/resourceGroups/rg1/providers/Microsoft.Network/dnsZones/zone1",
     "etag": "00000000-0000-0000-0000-000000000000",
     "location": "global",
     "name": "zone1",
     "type": "Microsoft.Network/dnsZones",
     "properties": {
       "maxNumberOfRecordSets": 5000,
       "numberOfRecordSets": 2,
       "nameServers": [
         "ns1-01.azure-dns.com",
         "ns2-01.azure-dns.net",
         "ns3-01.azure-dns.org",
         "ns4-01.azure-dns.info"
       ]
     },
     "tags": {
       "key1": "value1"
     }
   },
   {
     "id":
"/subscriptions/subid/resourceGroups/rg1/providers/Microsoft.Network/dnsZones/zone2",
     "etag": "00000000-0000-0000-0000-000000000000",
     "location": "global",
     "name": "zone2",
     "type": "Microsoft.Network/dnsZones",
     "properties": {
       "maxNumberOfRecordSets": 5000,
       "numberOfRecordSets": 300,
       "nameServers": [
         "ns1-02.azure-dns.com",
         "ns2-02.azure-dns.net",
         "ns3-02.azure-dns.org",
         "ns4-02.azure-dns.info"
       ]
     }
   }
 ]
}
```

And the method

```
func (client RecordSetsClient) ListAllByDNSZone(ctx context.Context, resourceGroupName
string, zoneName string, top *int32, recordSetNameSuffix string) (result
RecordSetListResultPage, err error)
```

Which lists all record sets in a DNS zone. Here is a sample request response

Request:

```
GET
https://management.azure.com/subscriptions/subid/resourceGroups/rg1/providers/Microsoft.N
etwork/dnsZones/zone1/all?api-version=2018-03-01-preview
```

Response:

```
{
 "nextLink":
"https://servicehost/subscriptions/subid/resourceGroups/rg1/providers/Microsoft.Network/dns
Zones/zone1/CAA?api-version=2018-03-01-preview&$skipToken=skipToken",
 "value": [
  {
    "id":
"/subscriptions/subid/resourceGroups/rg1/providers/Microsoft.Network/dnsZones/zone1/CAA/
record1",
    "etag": "00000000-0000-0000-0000-000000000000",
    "name": "record1",
    "type": "Microsoft.Network/dnsZones/CAA",
    "properties": {
     "metadata": {
       "key1": "value1"
     },
     "TTL": 3600,
     "fqdn": "record1.zone1",
     "caaRecords": [
       {
         "flags": 0,
         "tag": "issue",
         "value": "ca.contoso.com"
       }
```

```
      ]
    }
  },
  {
    "id":
"/subscriptions/subid/resourceGroups/rg1/providers/Microsoft.Network/dnsZones/zone1/A/rec
ord1",
    "etag": "00000000-0000-0000-0000-000000000000",
    "name": "record1",
    "type": "Microsoft.Network/dnsZones/A",
    "properties": {
      "metadata": {
        "key1": "value1"
      },
      "TTL": 3600,
      "fqdn": "record1.zone1",
      "ARecords": [
        {
          "ipv4Address": "127.0.0.1"
        }
      ]
    }
  },
  {
    "id":
"/subscriptions/subid/resourceGroups/rg1/providers/Microsoft.Network/dnsZones/zone1/CNA
ME/record2",
    "etag": "00000000-0000-0000-0000-000000000000",
    "name": "record2",
    "type": "Microsoft.Network/dnsZones/CNAME",
    "properties": {
      "metadata": {
        "key1": "value1"
      },
      "TTL": 3600,
      "fqdn": "record2.zone1",
    "CNAMERecord": {
     "cname": "contoso.com"
    }
   }
  }
 ]
}
```

These two methods can be leveraged to serve records from Azure. A **New** method can be used to validate the existence of domain name and hosted zone id key value pairs read from the Corefile and appends the vaild zones to a map.

The plugin would then run in an infinite loop and keep updating the zone information by requerying the resource record sets and updating the zone object

```
func (az *azure) updateZones(ctx context.Context) error {
    for zName, z := range az.zones {
        // .........

        // RS being the RecordSetsClient struct
        az.client.RS.ListAllByDNSZone(...)

        // .........
    }
}
```

The last phase would be focussed on writing documentation and unit tests for the plugin and achieving as much test coverage as possible.

Timeline

Pre-Community Bonding Period:

Make as many contributions to the CoreDNS organization as possible. Read up more on DNS server.

Community Bonding Period (April 23rd - May 14th):

Keep making contributions to the core repo. Try writing a few CoreDNS plugins of my own to get more comfortable with the project

Coding Phase (May 27th - Aug 26th):

Coding Phase 1(May 27th - June 24th):

Week #	Tasks	Deliverables
Week 1 & 2 (May 27th - June 9th)	● Write a basic plugin to read the azure config from the Corefile	● The plugin can parse the azure blocks in the Corefile
Week 3 (June 10th - June 16th)	● Implement chain credentials mechanism in the plugin ● Write the phase 1 blog post	● The plugin can parse the creds and initialize an authenticated client
Week 4 (June 17th - June 23rd)	● Buffer week ● Contribute to core repo	●

Coding Phase 2(June 25th - July 22nd):

Week #	Tasks	Deliverables
Week 1 & 2 (June 29th - July 12th)	● Work on the update zones logic	● The design and implementation details of the update mechanism are finalized
Week 3 (June 13th - July 19th)	● Implement fetching zones and record sets from Azure ● Write the phase 2 blog post	● The plugin is able to fetch resource record sets and updadte zones from Azure
Week 4 (July 19nd - July 22nd)	● Buffer week	●

Coding Phase 3(July 9th - August 6th):

Week #	Tasks	Deliverables
Week 1 & 2 (July 27th - Aug 9th)	• Write tests for the whole plugin • Try to achieve full code coverage	• The unit tests are written and code coverage is 100%
Week 3 (Aug 10th - Aug 16th)	• Document the plugin • Write the phase 3 blog post	• The plugin has been documented along with examples
Week 4 (July 17th - Aug 26th)	• Write the project report	• Final report submitted

Possible Outcomes:

- Coredns is able to serve zone data from Azure backends

Stretch goals / Future plans:

- I would like to keep contributing to CoreDNS as I am interested in making a career in the DevOps/Distributed Systems space and I think contributing to a CNCF project would help me gain the necessary insight and skills for this.
- Currently I am only trying to make PRs but I would also like to extend my contributions to raising issues and reviewing other PRs as I was doing with my previous GSoC community.

Why are you the right person to work on this project?

- I have ***past software development experience*** and have been working as a backend developer intern at http://appbase.io/ ***writing Go*** for about 6 months now so I am pretty comfortable with Go
- Last year ***I had completed a GSoC with coala.*** Here is my project completion report. So I am ***used to meeting the evaluation deadlines*** and ***submitting quality PRs***

- Have already started exploring the coreDNS project and I have ***made three*** **PR** to the core repo under review and been also trying my hand in ***writing a plugin*** by hacking on the demo plugin by Yong Tang. [Link to the plugin.]
- Have ***been contributing to open source software for over a year*** now so I am ***familiar with the git-flow*** and the best practices followed by various orgs.
- Have been researching and documenting my findings about my project and DNS servers in general since the last month here.
- The most important reason would be my love for open source software and communities and the desire to become a long term contributor to the communities that I contribute to.

Open Source Contributions

I am a regular open source contributor and have contributed to these organizations/projects (PR links are included) coala, Google, Ethereum Foundation, Kubernetes, Prometheus, Appbaseio, TaskCluster(Mozilla), DuckDuckGo, Kinto(Mozilla), Elastic, Kong, CoreDNS, Snowplow Analytics, NodeJS, OpenEBS and gojektech

Other Commitments

- **Do you have any other commitments during the GSoC period, May 8th to August 29th?**
  I have applied for a an internship at a startup. If selected I would like to do it
along
  GSOC. I know that I will be able to handle the two together because last year
  I had almost 30-40 hrs of free time even after working on my project. I have
  successfully completed a GSoC so I understand the pressure of meeting
  deadlines and can handle it well

- **Do you have exams or classes that overlap with this period?**
  No exams or classes during the coding phase.

- **Have you applied to any other organizations?**
  No I am only applying for the three coredns projects and no other organization.
My

Primary choice of project is the firewall plugin and the azure backend support is my secondary choice with google cloud plugin being my tertiary choice.