Connecting through a proxy

Connecting through a proxy is possible by:

- Using AgentProxy.
- Configuring `Client` or `Pool` constructor.

The proxy url should be passed to the `Client` or `Pool` constructor, while the upstream server url should be added to every request call in the `path`. For instance, if you need to send a request to the `/hello` route of your upstream server, the `path` should be `path: 'http://upstream.server:port/hello?foo=bar'`.

If you proxy requires basic authentication, you can send it via the `proxy-authorization` header.

Connect without authentication

```
import { Client } from 'undici'
import { createServer } from 'http'
import proxy from 'proxy'

const server = await buildServer()
const proxy = await buildProxy()

const serverUrl = `http://localhost:${server.address().port}`
const proxyUrl = `http://localhost:${proxy.address().port}`

server.on('request', (req, res) => {
  console.log(req.url) // '/hello?foo=bar'
  res.setHeader('content-type', 'application/json')
  res.end(JSON.stringify({ hello: 'world' }))
})

const client = new Client(proxyUrl)

const response = await client.request({
  method: 'GET',
  path: serverUrl + '/hello?foo=bar'
})

response.body.setEncoding('utf8')
let data = ''
for await (const chunk of response.body) {
  data += chunk
}
console.log(response.statusCode) // 200
console.log(JSON.parse(data)) // { hello: 'world' }

server.close()
proxy.close()
client.close()

function buildServer () {
```

```
  return new Promise((resolve, reject) => {
    const server = createServer()
    server.listen(0, () => resolve(server))
  })
}

function buildProxy () {
  return new Promise((resolve, reject) => {
    const server = proxy(createServer())
    server.listen(0, () => resolve(server))
  })
}
```

Connect with authentication

```
import { Client } from 'undici'
import { createServer } from 'http'
import proxy from 'proxy'

const server = await buildServer()
const proxy = await buildProxy()

const serverUrl = `http://localhost:${server.address().port}`
const proxyUrl = `http://localhost:${proxy.address().port}`

proxy.authenticate = function (req, fn) {
  fn(null, req.headers['proxy-authorization'] === `Basic
${Buffer.from('user:pass').toString('base64')}`)
}

server.on('request', (req, res) => {
  console.log(req.url) // '/hello?foo=bar'
  res.setHeader('content-type', 'application/json')
  res.end(JSON.stringify({ hello: 'world' }))
})

const client = new Client(proxyUrl)

const response = await client.request({
  method: 'GET',
  path: serverUrl + '/hello?foo=bar',
  headers: {
    'proxy-authorization': `Basic ${Buffer.from('user:pass').toString('base64')}`
  }
})

response.body.setEncoding('utf8')
let data = ''
for await (const chunk of response.body) {
  data += chunk
}
```

```
console.log(response.statusCode) // 200
console.log(JSON.parse(data)) // { hello: 'world' }

server.close()
proxy.close()
client.close()

function buildServer () {
  return new Promise((resolve, reject) => {
    const server = createServer()
    server.listen(0, () => resolve(server))
  })
}

function buildProxy () {
  return new Promise((resolve, reject) => {
    const server = proxy(createServer())
    server.listen(0, () => resolve(server))
  })
}
```